# DESIGNING, IMPLEMENTING AND EVALUATING, A FIRST STEP APPROACH FOR COMPUTATIONAL THINKING DEVELOPMENT IN NOVICE STUDENTS.

A dissertation submitted to Universidad de los Andes in partial fulfillment of the requirements for the degree of Doctor of Philosophy


Francisco Javier Buitrago Flórez

M.Sc., Universidad de los Andes

B.Sc., Universidad de los Andes

sicks@uniandes.edu.co

Faculty of Education – Universidad de los Andes, Bogotá, Colombia

Advisor:

Carola Hernández Hernández. PhD

c-hernan@uniandes.edu.co

Faculty of Engineering – Universidad de los Andes, Bogotá, Colombia

## Table of contents

# List of Tables

# List of Figures and Images

## *Figures*



## *Images*

# List of Appendixes

## Dedication

I would like to dedicate the development of this work firstly to my family. My Mom Adriana and my Father Francisco have been a central pillar for my development as a professional throughout my life; from the bottom of my heart I am sure I could not be able by any means to reach this level of academic development without them. Likewise, my sisters Andrea y Laura has been key encouragement and inspiration to reach this so special moment. All my family deserves as much recognition as I do for the development of this thesis since they were always extremely supportive with me -despite the distance- in critical moments of my life.

Secondly, I would like to dedicate this thesis to my mentors. My advisor Carola Hernández has been an immensely important person in the development of this Ph.D., leading to an overwhelming growth in my formation in the education field. I have to express that I consider I've learned a great amount of things under her tutoring, as well as increased in my professional skills thanks to her. On the other hand, Silvia Restrepo has been light and guidance during all my bachelor, master and doctoral programs. 12 years of tutoring and advising that in this very moment reach its highest point, profoundly influencing what I am now as a person and professional.

Thirdly, this document is dedicated to my closest friends. Alejandra, Andrés, Diego, Javier, Juan, Maria José and Robert heavily influenced the development of this path. By helping, cheering and supporting in several issues that surrounded the last 3 years of my life, they became in an important part of the development of this thesis.

**Acknowledgments**

I would like to thank to all the teachers and peers I had the opportunity to interact during this doctoral program since they provided marvelous experiences that contribute to my formation. Furthermore, I would like to thank to the Department of Science, Technology and Innovation of Colombia (Colciencias) for funding this project. Likewise, thanks to the Universidad Rey Juan Carlos, Madrid, Spain, for providing such a special experience during my internship. Special thanks to the thesis jury formed by Harold Castro Barrera Ph.D. and John Mendoza Garcia Ph.D. for being a critical part of this process. Additionally, I would like to thank to the students of the pilot program for offering themselves as volunteers; their work was undoubtedly a critical part of this research. Finally I thank anonymously, as requested, to all students' and the staff from the private school in Cajicá for providing the willing, the time and the facilities to carry out this research.

Francisco Javier Buitrago Flórez

Bogotá, Colombia

**“Knowledge management will never work until institutions realize it's not about how you capture knowledge, but how you create and leverage it.”**

**Etienne Wenger**

## Abstract

Computational thinking (CT) uses concepts that are essential to computing and information science to solve problems, design and evaluate complex systems, and understand human reasoning and behavior. This way of thinking has important implications in computer sciences as well as in almost every other field, since could be a source for critical skills development in students at any stage of education. As such, programming has been the main tool for CT development, usually engaging students in lectures and computational laboratory practices under the use of several programming languages such as C++, Java, Python and several others. Nevertheless, numerous issues associated with learning and teaching programming have been widely described. The overwhelming use of syntax and computational concepts, as well as issues regarding the use traditional-centered approaches for teaching and learning leads to a vast amount of students in failing to develop CT skills.

In this study, an innovate way for teaching and learning CT was designed, implemented and evaluated under the socio-cultural theory of education and the critical research method. On the one hand, the socio-cultural theory establishes that individuals interact in communities to build significant knowledge, meaning that learning must be seen as a social process. On the other hand, the critical research method indicates that pedagogical changes and reactions to particular educational issues must be processes mediated by the interaction and cooperation between the stakeholders.

Therefore, this study was framed under the research question “How a socio-cultural CT curriculum does influences the development of skills in novice students?” and was

subsequently divided in three phases: first, an in deep revision of literature was made to analyze and discuss findings regarding learning issues, as well as highlight the importance of learning programming focused on the development of CT skills at a young age. This activity lead to a description of the tools that are available to improve the teaching process of CT, as well as provided a state–of–the–art overview of how programming is being taught at schools and universities in Colombia and around the world. Likewise, as a result it was found that one of the main issues regarding the failing rates in initial programming courses relies in the minimum development students have of basic CT skills (abstraction, algorithmic thinking, decomposition, debugging and generalization), which are critical for the understanding of programming and further CT development.

In response, a set of active pedagogical strategies framed in a socio-cultural vision of education were designed and tested qualitatively in a pilot program, to offer an alternative way to immerse university students in the learning process of CT development. Results showed that the curriculum design engaged students in the active use of five key skills related to CT, which could be used as the base ground for further programming learning and high-level CT skills development. Additionally, the pedagogical approach indicates that students were involved in reflective process of learning, as well as in the development of key competencies such as teamwork, communication skills and creativity.

Given the results, the socio-cultural CT curriculum was improved to not only develop CT skills, but also critical thinking, communication, collaboration and creativity. These competences are well described in literature as key competences to face the challenges of a rapidly changing world, given the social, technological and economic changes in recent decades. Furthermore, an embedded mixed-method approach was

implemented to evaluate the students' improvements in competence development in both, quantitative and qualitative ways. The results showed an encouraging increase in skills related to the competences of interest thanks to the implementation of a student-centered pedagogical curriculum based on CT. Additionally, by designing the curriculum under socio-cultural ideas of education, results indicate the students and the teacher were able to form a community to facilitate teaching and learning as social processes.

In conclusion, I can argue that computational thinking holds the potential for key skills development in students, as is an effective way to approach and solve problems not only related to computer science and applications, but also problems related to other fields and every day situations. Moreover, by implementing and evaluating novel approaches that relies in active interactions between students and researches, as the socio-cultural vision of education proposes, learners would be able to move forward skills and competences development instead of memorizing concepts. Investigation regarding this sort of curricula leads to the discovering of innovative pedagogical approaches, which undoubtedly serve as a response to the educational challenges universities and schools are facing towards the development of fully equipped citizens for a 21$^{st}$ century world.

# Chapter I

## Introduction

Computational Thinking (CT), as defined by Jeannete Wing in 2015, "*is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent*" (Wing & Stanzione, 2016). The concept and definition of CT have been a topic of discussion and research in the past decade, due to the capacity of CT to be used as a tool to develop skills of high level of complexity in students ranging from K-12 to bachelors' programs and beyond (Bocconi, Chioccariello, Dettori, Ferrari, & Engelhardt, 2016; Buitrago-Florez et al., 2017; Curzon, Dorling, Ng, Selby, & Woollard, 2014; French Academy of Sciences (FAS), 2013; Qualls & Sherrel, 2010; Wing, 2006b).

Several lines of evidence exist to support the implementation of computational thinking in schools as well as in higher education. First, children are now more than ever exposed to considerable amounts of data, sometimes not curated, delivered in several formats, and most of the time, not interconnected. CT can provide children with the tools to extract the knowledge that might be hidden in the data. Second, we are used to teaching reductionist science (i.e. isolated mechanisms or metabolic pathways) but need to start teaching the systems as they are: complex. Most of the times, because of the lack of time, teachers do not ask or guide their students to make connections between the materials they receive. Third, CT helps to enhance the thinking abilities as we can superimpose and combine multiple layers of abstraction as a computer programmer does when developing algorithms (Buitrago-Florez et al., 2017).

Moreover, CT not only prepares students in the field of computer science, but also provides students with tools and skills to approach and solve a wide range of problems in different areas of knowledge (Werner, Campe, & Denner, 2012). Nowadays, CT influences fields related to Natural and Social Sciences. It allows the acquisition of skills to solve problems, design systems, and understand the power and limits of human and machine intelligence. CT has been claimed as a way of thought that every person must be able to perform, since provides a key problem solve strategy and develops high level thinking skills (Wing & Stanzione, 2016). Curzon et al. (2014) highlight 5 essential skills related to CT that students should be able to display, described as follows:

- Abstraction: It's a different way of problem approach that simplifies the situation. Fundamentally, involves the process of hiding unnecessary details of the situation in order to solve the problem in a most efficient way. The ability resides in being able to choose properly which details must be hidden and which details must remain relevant to solve de problem.
- Algorithmic thinking: It´s a way of thinking that enables a person to formulate a solution by a set of clear steps. Basically, the student develops a set of instructions or rules that provides a clear way to solve a specific problem.
- Decomposition: It´s a way of thinking about problems and solutions in separated components. A problem can be divided into separated parts that can be understood, developed, solved and evaluated separately, making complex problems or situations easier to solve.
- Debugging: A systematic approach to find and resolve defects in the proposed solution. Involves the removal of unnecessary steps in order to increase

efficiency and effectiveness.

- Generalization: It's a way in which new problems can be solved based on solutions previously created. The student can take the solution of a previously solved problem and adapt it in such a way that it can solve not only one, but a whole class of similar problems. Thus the student would be able of reducing the use of time and economic resources.

On the other hand, Wing's (2011) definition of CT is comprised by three main characteristics. First, CT is a thought process independent of technology. This suggests that to teach or learn CT, computers are not necessary. Furthermore, CT is a problem-solving approach that represents solutions that can be carried out by a human, a machine, or a combination of both, meaning that the solution does not necessarily has to be processed by a computer. Second, the use of CT involves a set of high-level skills used to identify and solve a problem. It is worth noting that the term CT may be sometimes used indistinctively from Algorithmic Thinking. However, it is important to view these two as different concepts, where CT is a powerful way of thinking for problem solving, whereas algorithmic thinking is a specific skill to be used in CT (Syslo, 2014).

Third, programming has been referred to as the best method to teach CT. Programming is the process in which a person is able to provide a set of instructions that will communicate, as specific and accurate as possible, a procedure, method, practice, or task to a machine. This set of instructions must be coded using a specific programming language (Vihavainen, Airaksinen, & Watson, 2014). However, most of the programming courses are initially taught by introducing the student to a coding language in order to solve a problem, which will be carried out by a machine (Bennedsen & Caspersen, 2007).

Unfortunately, this approach presents several problems. On the one hand, students that take a programming course are given tasks that require a set of abilities for solving problems of exercises in class (CT skills). The problem is that teachers and professors sometimes assume students have these abilities, although it is not the case most of the time (Lahtinen, Mutka, & Jarvinen, 2005). On the other hand, students must learn a specific programming language to solve their in-class problems and exercises, leading to an overwhelming number of syntaxes and semantics as a consequence of the diversity of commands that are specific to each programming language (Jenkins, 2002). Both of these tasks are high-level processes that might provoke high stress to students (Grandell, Peltomaki, Back, & Salaskoski, 2006a), and can be compared to learning math concepts in an unknown, foreign language.

To ease the challenges students face while learning to program and develop CT, several approaches like video games, specialized software, LEGO Robots suites, and specialized web sites such as Code.org have been developed (Buitrago-Florez et al., 2017). Unfortunately, regardless of all the efforts done to overcome the difficulties associated to learning programming an important number of students are challenged by abstraction, algorithmic thinking, decomposition, debugging and generalization (Curzon & Mcowan, 2017). CT must be seen as a framework to develop concepts and skills related to computer science, rather than a human–machine communication tool, related to the information and communications technology field "commonly referred as ICT" (French Academy of Sciences (FAS), 2013). Thus, instead of teaching students solely how to write code, CT courses must include within its learning objectives the development of skills related to CT. Therefore, students exposed to CT could develop some of the most powerful CT skills that

may be applied in a variety of real world problems in different contexts, and could be engaged into the development of additional skills via further computer programming courses (Butler & Morgan, 2007).

Consequently, the purpose of this doctoral thesis is to produce an in-deep review of the literature regarding all the efforts established on programming courses to develop, implement and evaluate a computational thinking course to foster skills and competencies in novice students. Thus, this study is framed under de following research question:

> How a socio-cultural CT curriculum does influences the development of skills in novice students?

Then, the general objective of this doctoral program is:

> Design, implement and evaluate a socio-cultural CT curriculum aimed to the development of key skills in novice students.

The general objective is subsequently divided into four specific objectives:

1. Realize a state of the art in Teaching and learning programming to develop CT.
2. Analyze the differences between Computational thinking and programming as a theoretical reference of curriculum design.
3. Design, implement and assess (pilot) a computational thinking curriculum for novice students.
4. Perform a Program evaluation of computational thinking curriculum (1 cohort).

# Chapter II

# Methodology

This section summarizes the efforts to successfully develop, implement and assess a computational thinking curriculum approach. As such, this chapter covers the description of the educational theory used as the framework for the curriculum design, as well as the educational research strategy that encompasses the implementation and evaluation of the courses developed in this project.

## A socio-cultural theory of education for teaching and learning CT

Socio-cultural approaches in education claims that learning is a complex problem which is the product end of different activities, contexts and socio-cultural factors involving the learner (Vygotsky, 1978). As such, it is plausible to state that the origins of human thinking may be inseparable from social and cultural practices (Hernández, Ravn, & Valero, 2015). This perspective of learning, as discussed by (Radford, 1997), proposes that knowledge is the product of negotiation of meaning, framed by activities of individuals in a cultural context. In other words, learning occurs as a specific social process, in which the learner becomes progressively expert in cultural forms of thinking through mastering of language, interactions, signs and artifacts (Radford, 2008). This vision of learning could be easily seen when a professor complains about the inexperience or lack of knowledge of new graduate students. However, after a few years of working in the specific field, professors realized that students are transformed by learning in context (Wieman, 2007).

Embracing a socio-cultural perspective in education research indicates that researchers and practitioners must envision education processes as social activities

developed in communities (Leach & Scott, 2003). Therefore, learning must not be considered as an isolated and individual process, on the contrary, learning must be understood as a distribution and transformation among members of a community of practice (Wenger, 1998). For Wenger, the term community of practice derives from the notion of learning as active social participation, and lies in the idea that the process of learning occurs when people who have a common practice share ideas, values, beliefs, languages and ways of doing in an extended period of time (Hernandez, 2015).

From a socio-cultural perspective of education, students could get involved in authentic learning experiences through the immersion in routines, rituals and conventions of their professions. This process allows the learners to acquire skills, language, values and knowledge associated with their specific practices (Sutherland, Scanlon, & Sperring, 2005), taking into account that the way in which members of specific professions interact and the way they use language, sign systems and tools is particular for each career (Northedge, 2002). Consequently, the learning process can be seen as the learner moving from the periphery of a practice community all the way to its center, where each step transforms the practitioner into a member of this community as s/he learns and participates in each step taken along the way (Hernández et al., 2015).

Hence, the development of the computational thinking skills and competencies could be seen as a social process, in which the learner transits a path framed by activities of individuals in a context. Along this path, the student enters in a community of practice as described by Wenger (1998), in which a member who gets involved in participation (actively interacting and creating identity in the community) and reification (transforming abstract information into real artifacts) progressively becomes an expert. Thus, the learning

process can be seen as the travelling of the learner from the periphery to the center of a community of practice of CT, transforming the practitioner into a fully equipped individual who can interact with its society (Hernández et al., 2015).

This vision of education allows students to be the center of their own active learning process, involving the learners in actual activities of the community of learning. This vision of learning is far different from the traditional teacher-centered strategies, in which teaching is seen as a process of transmission form the teacher to the students not only of information, but also concepts, skills and competencies (Roth & Lee, 2004). Thus, in student-centered strategies, students are expected to be involved in experiences that lead to negotiation of meaning, facilitating the formation of an identity inside the community from which the teacher belongs too. In this sense, the role of the professor changes from the "source" of a knowledge to be supposedly transmitted, to an experienced member of the community who helps the students in process of becoming active members of the it (Radford, 2008).

Additionally, the socio-cultural vision of education allows students, teachers and researchers to perform valuable reflection processes to ensure the quality of learning and teaching activities. On the one hand, students are encouraged to engaged in reflection and conscious awareness processes, allow them understand what may have done wrong in order to avoid future pitfalls, as well as understand the objectives of learning of the activities they perform (Turns, Sattler, Yashuhara, & Borgford-Parnell, 2014). On the other hand, teachers are able to carry out research of their pedagogical strategies allowing all members of the communities of learning to interact and participate in the development or improvement of pedagogical strategies (Agouridas & Race, 2007).

## Critical Research for CT development

This doctoral research is formulated under the basis of the critical research proposal made by Skovsmose and Borba (2004). From the authors' perspective, critical research is a model of investigation by which investigators are able to perform transformations in the pedagogical practice through cooperation between students, researchers and teachers. The main characteristic of this model resides in the ability of solve a specific problem taking into account that the situation does not occur in a specific moment in time, but can reappear after the implementation of a specific action. From this vision, an educational process of transformation would be represented through three analytic situations: The *current situation*, the *imagined situation* and the *arranged situation*.

As proposed by Skovsmose and Borba (2004), the current situation is the starting point of the research process. The current situation is identified through an in deep analysis from the specific actors (researchers and teachers) belonging to the educational context subject of intervention and change. Furthermore, the imagined situation arises from a reflection of the current situation. The imagined situation is characterized by expectations, possibilities and longings for change given the current situation. Finally, as researchers and participants of the investigation process go forward in development and implementation of the imagined situation, alternatives emerge from negotiation processes among those involved. These alternatives are therefore converted into specific actions, which are adapted in the research scenario and falls into what the author's call the arrange situation. Figure 1 represents each of the situations described by (Skovsmose & Borba, 2004).

From this perspective, all the situations are critical gears of the expected transformation. Consequently, researchers are encouraged to focus not only in the current situation, but to develop several imagined situations to solve the problem in question. To fulfill this idea, it's necessary from all participants to confront what is happening in the reality with what could happen in the future.

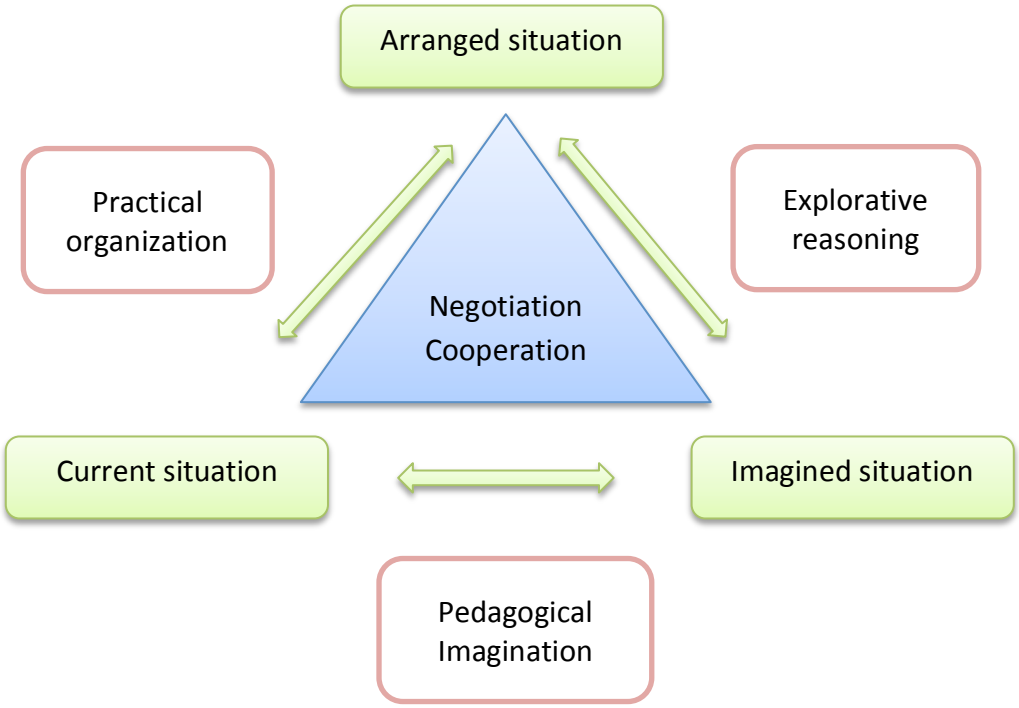



Figure 1. Analytic representation of the critical research model stated by Skovsmose and Borba (2004)

Skovsmose and Borba (2004) described three different negotiation processes that require different levels of cooperation between students, teachers and researchers. As seen in Figure 1, in the middle of the current situation and the imagined situation lies the pedagogical imagination. The pedagogical imagination is described by the authors as the actions and conceptualizations that allow all participants to negotiate and built several

alternatives to change the current situation, by using at least two specific resources: the practical knowledge from the teachers and the theoretical approaches from the researches.

The second negotiation process, called practical organization, lies between the current situation and the arranged situation. This process gathers all the activities needed to perform a change situation towards the imagined situation. It's important to mention that usually, the arranged situation differs from the imagined situation, due to negotiation processes between all participants in the practical organization. Finally, the third negotiation process is the explorative reasoning, which derives from an analytical reflection of the imagined situation in contrast of experiences given by the arrange situation. This explorative reasoning is a critical interaction between the pedagogical imagination and the practical organization.

Consequently, it is possible to think that pedagogical changes in classroom practices are developed as a movement of these three situations through time. In other words, the current situation it is established to be altered, as well as the imagined and the arranged situations. Therefore, once the process of critical research recovers for a first time the triangular path, in the end the arranged situation could be seen as the current situation for a second iteration of research, which leads to further imagined and arranged situations (Figure 2). These iterative processes could be developed as many times as necessary to fully generate a meaningful change in pedagogical practices in a classroom.

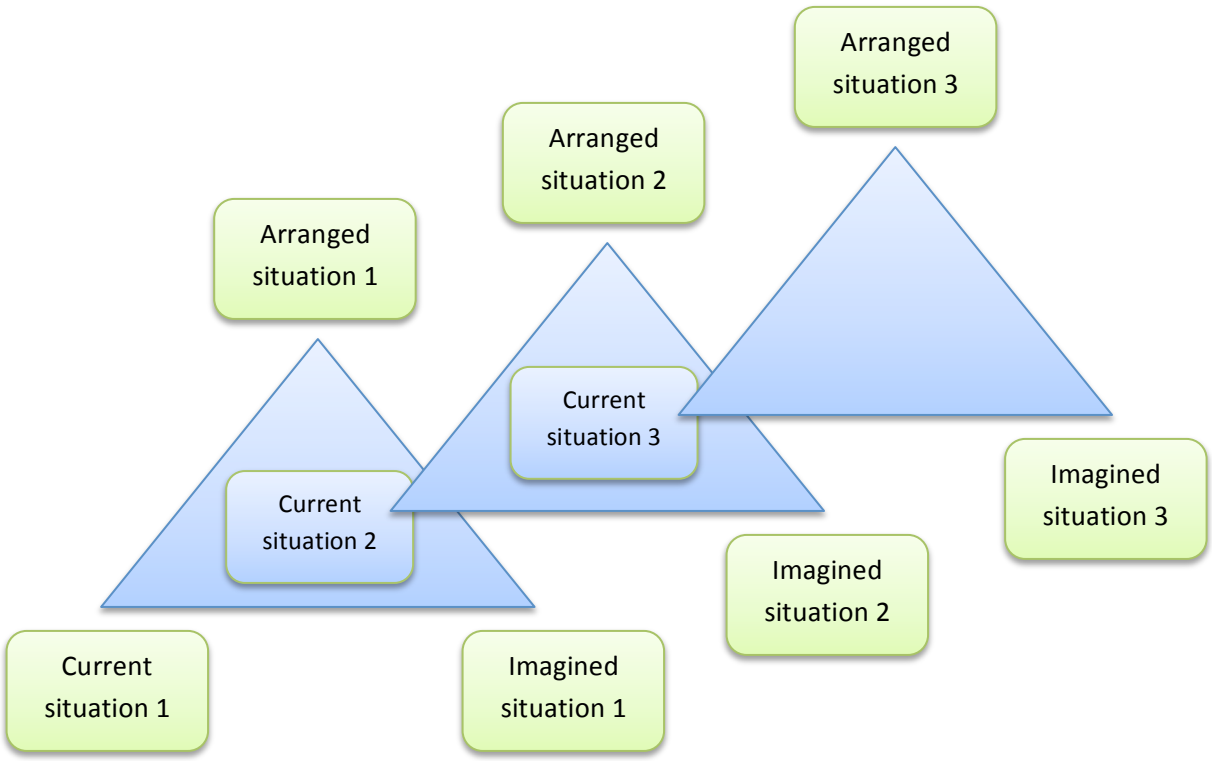



Figure 2. Iterative representation of changes in the classroom under the critical research model stated by Skovsmose and Borba (2004)

Taking into account the iterative model, this doctoral investigation was developed through two triangular paths of critical research. Considering the first path, the *current situation 1* derived from the results of the literature revision of approaches related to development of computational thinking through programming. Therefore, a pedagogical imagination process produces the *imagine situation 1,* which was a CT curriculum design based on active pedagogical strategies supported by the socio-cultural vision of education to develop key skills in novice students. Consequently, the *arranged situation 1* was the conglomerate of experiences product of the practical organization of a pilot program of the CT curriculum.

This situation subsequently became in the *initial situation 2* through explorative reasoning of the results of the pilot program. Hence, given the outcomes from the first phase, a second process of pedagogical imagination leads to the *imagined situation 2,* which was a progression in order to strengthen the CT curriculum, to be later implemented in a full CT course. Afterwards, a second practical organization produced the *arranged situation 2,* in which the experiences of the participants were gathered to generate a new event of explorative reasoning, in which results were evaluated to reach conclusions about this pedagogical innovation (Figure 3).

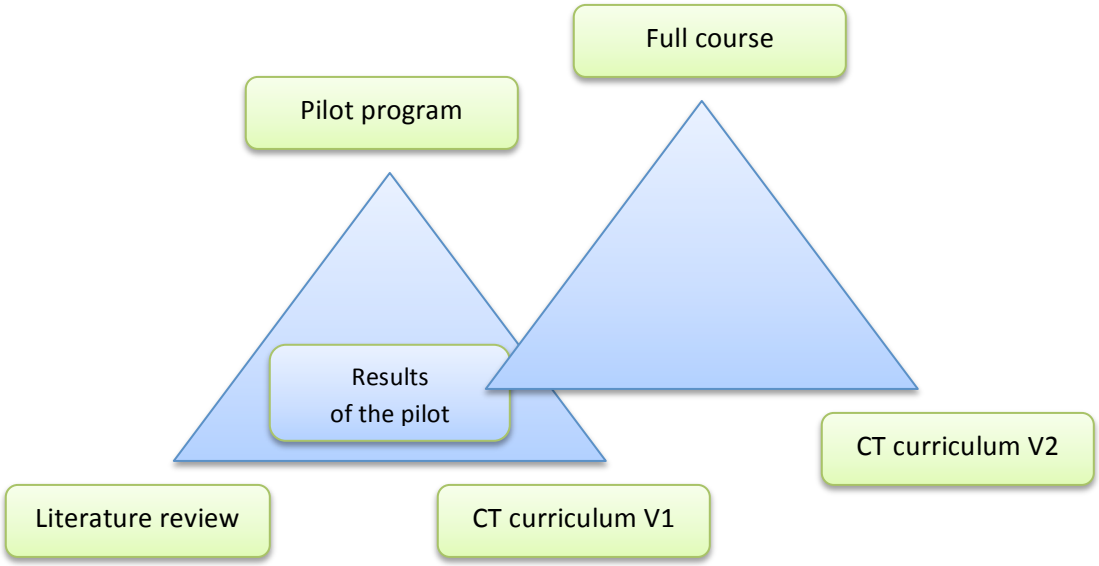



Figure 3. Iterative representation of this proposal under the critical research model stated by Skovsmose and Borba (2004)

Each of the steps made throughout the triangular paths were developed under particular methodologies for data gathering and analysis, in order to maximize the amount and quality of the information. As such, these methodologies were established and used

consistently with the epistemological considerations proposed by the critical research, in which the intervention and collaboration between researchers, teachers and students, must rely in principles of validity, reliability, legitimation and justification to perform a comprehensive evaluation and derive accurate conclusions.

As stated by Stake (2004) comprehensive evaluation is more an attitude than a recipe; hence, this kind of evaluation is, at great extent, based in an interpretative way of thinking. The merit and worth of something related to evaluation is usually done after successive debugging of personal judgment and not by comparing a numerical performance indicator with a standard. Consequently, since I was the designer, the teacher and the evaluator of the program, my role consisted in being one of the voices (highly influenced and far from being objective) of judgment of the study, since the process of judging program quality involved a thoughtful dialogue among diverse stakeholders (Green, Bouce, & Ahn, 2015).

Following the methodological aspects of the critical research, negotiation processes occurred between teachers and researchers in the design of material for the Problem-based activities and the elaboration of the instruments for data collection. The ones involved in these processes were, my advisor, other CT researchers and teachers at Universidad de los Andes and I as researcher and teacher. Quantitative and qualitative instruments were design throughout a doctoral class project in which experts in instrument design and data gathering guided the development to provide validity and legitimation to the process. Likewise, it was my function as evaluator to provide validity to the quantitative embedded approach, by assuring objectivity in the design, implementation and evaluation of the pre/post-tests for the mixed-method approach. Additionally, throughout the development

of the pilot and the full course of CT, students were able to interact and participate directly by giving feedback of the curriculum in written reflections and focus group, allowing them to being active in the investigation process as described by Skovsmose and Borba (2004)

In terms of the evaluative logic behind this evaluation process, the Figure 4 describes how antecedents, transactions and outcomes were addressed in the comprehensive evaluation. As stated by Stake (2004), antecedents are any condition existing prior program implementation that may lead to outcomes; transactions are the countless encounters between students and teacher, students and other students, teacher and researchers and between other actors involved in the process; finally, outcomes could be measurements of the impact and/or perception among stakeholders.

Accordingly, all instruments used for qualitative data gathering were carefully chosen to provide enough narrative data for a triangulation process. The use of written reflections, field dairy of the teacher and focus group, in combination with the discussions with my advisor and other teachers involved in CT regarding instruments and results associated, was intentionally made to provide justification to judgments from data. Consequently, the claims regarding the development of CT skills and competences previously stated are legitimated, and show the quality of the CT curriculum evaluated.

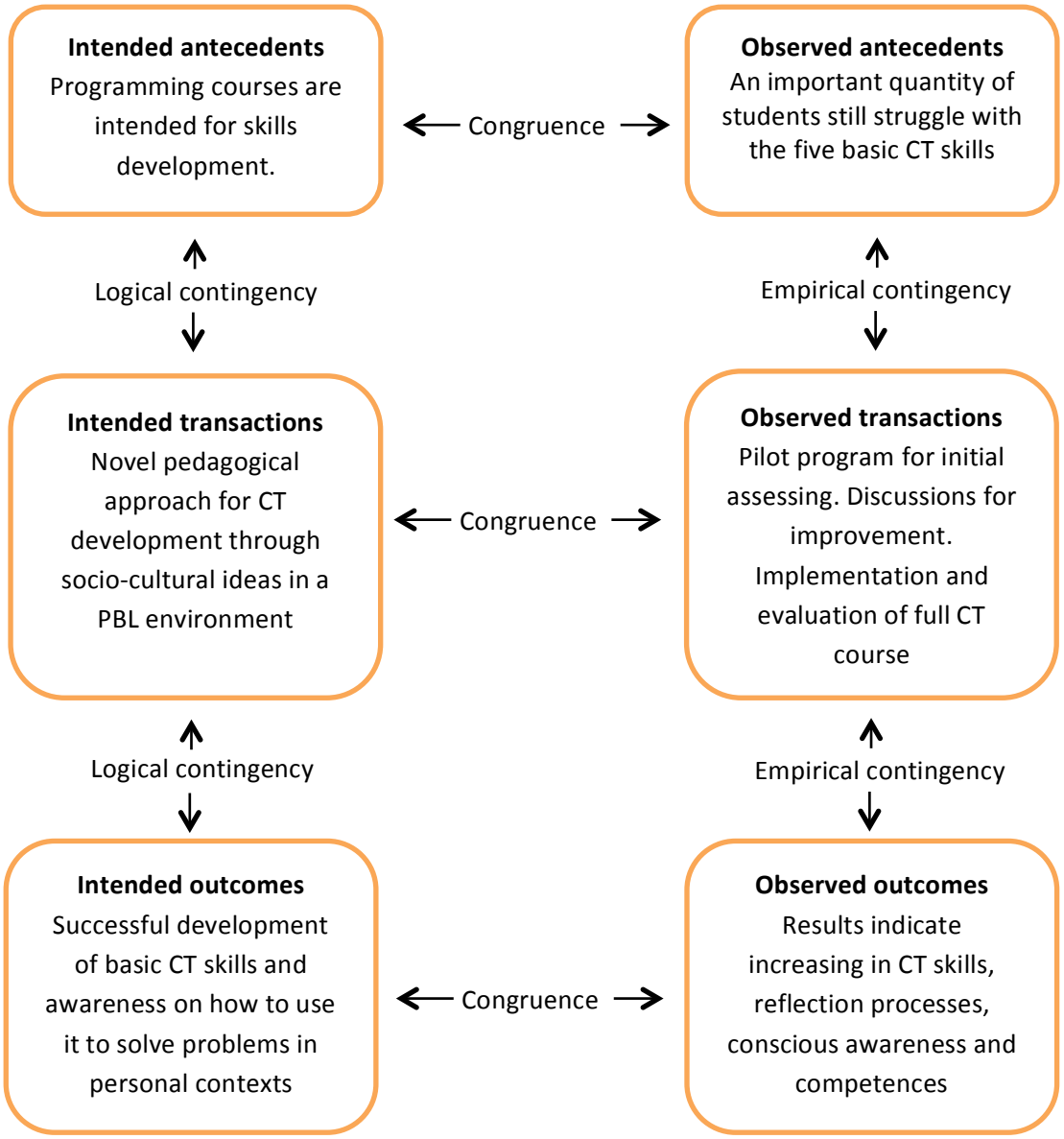



Figure 4. A representation of the processing of descriptive data (modified from Stake 1967)

Additionally, this proposal was submitted to the ethical committee at the Faculty of Education of the Universidad de los Andes and was, in each of its phases, evaluated and successfully endorsed as a study of minimum risk. Informed consents from all participants were used and anonymity, protection and privacy of the data recollected were ensured in each of the stages proposed in this document.

Taking into account the critical research steps described above, the following chapters III and IV explore the development of the first triangular path of critical research in the development of a curriculum to immerse novice students in CT.

# Chapter III

## Defining the first current situation

Taken and adapted from "Buitrago-Florez, F., Casallas, R., Hernandez, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a Generation's Way of Thinking: Teaching Computational Thinking Through Programming. *Review of Educational Research, 87*(4), 834-860".

Given that programming has been used in an extensive period of time as the main tool for skills development, it is necessary to understand the state of the art in the teaching and learning processes of it. Therefore, this chapter explores the history of the development of numerous pedagogical strategies designed to overcome the difficulties associated to development of CT through programming in higher and school education, at the same time that sheds light of a critical concern that leads to the *current situation 1* of this critical research process.

To obtain an overview of the state of the art of teaching and learning programming in educational institutions, a worldwide extensive search of papers in journals and proceedings of the Association for Computing Machinery Digital Library, as well as in journals specialized in education of computing or programming (i.e. Review of Educational Research, Computer Science Education, Education and Information Technologies, Computers in Mathematics and Science Teaching, Journal of Virtual Worlds Research, Educational Technology & Society, among others) was made. Several key words and key phrases were used throughout our search. Among these were: "teaching/learning programming", "issues in teaching/learning programming", "challenges for novice programmers", "programming in schools", "programming in higher education", "coding for children", "woman in computing", "coding in the 21st century", "programming in the

United States/Europe/Asia/Latin America", "approaches to teaching/learning programming", "collaborative teaching/learning in programming".

Subsequently, a search for literature that might not be published in indexed journals or proceedings was made. An open search in Google Scholar using the same key words and key phrases, to gather valuable information from associations of teachers and individual initiatives in the field. Given that these references were from unindexed sources, it was decided to restrict the search to associations of teachers in leading countries in this area. Besides, information from non-profit organizations that aim to encourage the teaching and learning of programming at any age was gathered.

The process ended up with 92 references including journal articles, reviews, proceedings, short communications, and governmental standards from established associations of teachers in the United States, UK, France, and Germany. No restrictions regarding the date of publication were taken into account. However, information published in the last 15 years was the most relevant for this research.

**Teaching and Learning Programming in Higher Education**

Over the last decade, computer science and programming have grown remarkably given that these disciplines are permeating several fields such as biology, chemistry, physics, medicine, engineering, art, music, and social sciences. Furthermore, new subfields such as bioinformatics, internet security and ethics, gaming, artificial intelligence, among others have been created (Kay, van der Hoek, & Richardson, 2005). As shown in Table 1, computer programming focused in the development of CT skills is an ability that is very useful and can be very rewarding for students in numerous disciplines (Piteira & Costa,

2012). Teaching programming languages to foster CT in higher education however, has proven to be an extremely difficult and challenging task for both students and professors (Mow, 2006). Therefore, experts in this area have been analyzing the difficulties faced by novice students taking introductory programming courses, particularly due to the fact that programming places a heavy cognitive load on freshmen students (Ahmadzadeh, Elliman, & Higgins, 2005; Flowers, Carver, & Jackson, 2004). It is important to clarify that most freshmen students have had no programming background knowledge or experience (Tuugalei & Mow, 2012). According to Flowers et al. (2004), even after two years of learning programming, most of the students are still struggling to be proficient.

**Table 1**

*Fields in which Programming and Computational Thinking can be directly applied*

| Area | Activities related to programming and CT |
|---|---|
| **Biology** | Genome sequencing, genome assembly, and gene prediction<br>Protein interaction modeling<br>Metabolic pathway reconstruction<br>Population and systems biology |
| **Chemistry** | Discovery and design of drugs<br>Molecular dynamics simulations<br>Chemical pathways modeling<br>Study of fundamental properties of atoms and molecules |
| **Physics** | Physical behavior of materials<br>Optical performance simulations<br>Astrophysics modeling<br>Physical interaction in biomolecules |
| **Medicine** | High–throughput biomedical assays<br>Neuronal pathways modeling<br>Physiology performance simulations<br>Medical surgery automation<br>Analysis of drugs and environmental toxins<br>Medical illustration |

| | |
|---|---|
| **Engineering** | Mobile and internet computing<br>Gaming<br>Robotics<br>Computer and network security<br>Artificial intelligence<br>Data base systems |
| **Arts** | Image design<br>Movie animation |
| **Music** | Acoustics simulations<br>Recording techniques<br>Sound synthesis and manipulation<br>Computer music |
| **Social Sciences** | Demographic simulations<br>Pandemic reaction modeling<br>Computational finance |

Programming requires memorizing a wide range of information and displaying several skills at the same time. A student must think about the details of syntax and semantics of the programming language used, elaborate some mental model of how to solve each problem, and be able to distinguish between solving the problem and specifying the solution in a way that a computer would be able to execute it (Pane & Myers, 1996). Consequently, a student must be capable of using abilities related to computational thinking such as algorithmic thinking (in order to obtain a solution through clearly defined steps), evaluation (ensuring that an algorithmic solution is a good one), decomposition (split the situation in several parts), and abstraction (in order to basically make problems or systems easier to think about, removing unnecessary complexity and details (Curzon et al., 2014).

Numerous problems associated to the learning of programming have been largely described for several years. Lahtinen et al. (2005) stated that novice programmers are

typically limited to the surface knowledge of programs, often getting a programming approach "line by line" instead of using meaningful program structures using CT skills. This, most likely leads to a failure in terms of applying the knowledge they have obtained satisfactorily. Additionally, it seems that students may know the syntax and semantics of individual statements, but they do not know how to combine them into valid programs as they show a lack of detailed mental models and are thus unable to solve problems in an efficient way (Winslow, 1996). Moreover, most students present several issues when they have to understand notions about loops, conditionals, arrays, and recursion (Robins, Rountree, & Rountree, 2003), as well as pointers and references, finding bugs from their own programs, structured and abstract data types, and error handling (Lahtinen et al., 2005; Piteira & Costa, 2013; Tuugalei & Mow, 2012).

In regards to these problems, diverse strategies have been tested in freshmen students to confront the adversities associated with learning to program. After an extensive survey conducted to both students and professors, Lahtinen et al. (2005) found that practical sessions in computer rooms were highly rated as the most useful strategies. In the Internet, students encounter a significant number of examples of codes (executable), programming tutorials, educational videos, and contents available on educational platforms such as Moodle and Blackboard (Piteira & Costa, 2013).

Furthermore, Vihavainen et al. (2014) conducted a literature review to evaluate different worldwide teaching intervention approaches to address the previously mentioned problem. The authors grouped all interventions into four different categories. The first category was Collaboration and Peer Support. This includes: (a) Peer–Led Team Learning activities in which a team based learning is provided to help the students in introductory

programming courses (Lasserre & Szostak, 2011); (b) Pair Programming Activities in which two students, with specific roles, work collaboratively at one computer on the same design, algorithm, code, or test (Williams, McDowell, Nagappan, Fernald, & Werner, 2003); and (c) Cooperative and Collaborative Practices in which an undergraduate student co–teaches the class as the students are organized in cooperative groups (Chase & Okie, 2000). The second category was named Relatable Content and Contextualization. This includes: (b) Media Computation in which design, audio, and Web page programs are used to facilitate teaching programming (Tew, Fowler, & Guzdial, 2005); (b) Real World Projects in which a set of lab exercises based on real–world problems assist the teacher in the class (De La Mora & Reilly, 2012); and (c) Courses that evolve around games in which using game maker platforms students are encouraged to design a game as desired (Rankin, Gooch, & Gooch, 2008b). The third category was named Course set up, Assessment, and Resourcing. This includes: (a) Adjusting Course Content in which student's bottlenecks are analyzed and then contents are adjusted (Shaffer & Rosson, 2013) and (b) Changing the Grading Schema so that all assignments are evaluated in a way that would increase the pass rate and consequently generate a better atmosphere in the class (Nikula, Gotel, & Kasurinen, 2011). Finally, the fourth category was named Hybrid Approaches. This fourth category combines the methodologies applied in the previous three categories, including Media Computation with Pair Programming, Peer–Led Team Learning, and Collaborative Learning with Relatable Content.

At the end of their review, Vihavainen et al. (2014) claimed that on average, teaching interventions can improve programming pass rates by nearly one third when compared to a traditional lecture and lab based approach. Moreover, authors conclude that

courses with relatable content (e.g. media computation) and cooperative elements (e.g. pair programming) increase the pass rates and intrinsic student motivation, whereas courses that relay exclusively on a single approach performed poorly.

As a first hand case I want to highlight a specific example at Universidad de los Andes (Bogotá, Colombia). Since at least a decade ago, the Systems and Computing Engineering Department has been conducting surveys and working on some of the most common problems related to teaching and learning programming. Among those identified it's important to highlight: (a) motivational problems, finding that some students have the impression that success depends on something beyond their control regardless of the understanding of the theory, algorithmic skills, or the time spent working on a program; (b) lack of congruence between the concepts and skills taught in introductory courses and the ability to write specific programs to solve a problem in an efficient way; and (c) methodological problems in teaching programming. The instructor commonly assumes that students will be able to learn on their own by going over examples presented in class and are then expected to design their own programs (Villalobos, Calderon, & Jimenez, 2009; Villalobos, Casallas, & Marcos, 2005).

In order to prevail over the issues previously described, the Systems and Computing Engineering Department at Universidad de los Andes has been working since 2006 on a robust interactive platform called Cupi2 (http://Cupi2.uniandes.edu.co). This interactive platform embraces an extensive set of resources that include lectures, workshops, books, articles, lab examples, concept maps, tutorials for students and teachers, as well as reports about class schema and student performance on every single programming course taught at the university (Vega, Jimenez, & Villalobos, 2013). Cupi2 was born as an initiative to

tackle major challenges in teaching and learning programming. The main goal was to design learning strategies that use problem based learning (PBL) and bottom–up approaches to introduce simple concepts (data types, operators, and expressions) and then lead students towards more complex concepts such as control structures, methods, parameters, vectors, and collections in a Java environment (Villalobos & Casallas, 2006).

An introductory course to programming called APO 1 (algorithms and object–oriented programming) is taught every semester at Universidad de los Andes. This course uses the Cupi2 interactive platform and is taken by students in any of the undergraduate programs from the School of Engineering, the School of Sciences (except for students in biology and microbiology), and some students in the music department, reaching more than 1,000 students per semester. Each class has less than 26 students (to improve the one to one interaction between students and teachers) and is divided into six levels, which reflect the difficulty and skills required. In levels one to five, students are taught the main concepts related to object–oriented programming through lectures and examples included in the textbook written by professors associated to the Cupi2 interactive platform. Furthermore, students attend lab sessions where they have the opportunity to practice problems that are partially solved, in a java environment called Eclipse (http://www.eclipse.org/), using the resources available in the Cupi2 platform. Through *CupiTaller*, which stands for "CupiWorkshop", students from all academic programs taking APO I or APO II (the course that follows APO I) may access a physical space where they may receive personalized tutoring to strengthen their programming abilities. At the end of the course (level six), students are challenged to solve a complete exercise using all the tools, syntax, and skills learned throughout the semester. These efforts have shown an

improvement in the success rates and motivation of the students. A nearly 50% decline in the number of students failing the course has been observed as well as a consistent reduction in the number of students that drop the course. Moreover, questions related to the student's perception of the course showed an overall 20% increase after the introduction of Cupi2 (Villalobos et al., 2005). Around 25 universities in Colombia work with and develop materials for Cupi2 each year, creating a large and diverse community for teaching and learning programming (Villalobos et al., 2009).

Unfortunately, regardless of all the efforts done to overcome the difficulties associated to learning programming (i.e. Cupi2), an important number of students are still challenged by abstract programming concepts and logical reasoning. These are some of the most powerful CT skills that may be applied in a variety of real world problems in different contexts (Butler & Morgan, 2007). An important number of students and professors at Universidad de los Andes feel that most novices are able to understand the majority of concepts, but are unable to apply them correctly, which is consistent with the findings of Bruce (2005). When reviewing the end of semester surveys of the APO I course, students from the first level particularly stated that they could have learned more if they would have had prior experience with programming before taking this first–semester course. Despite the success of Cupi2 in helping students deal with several issues associated to introductory programming courses, the lack of expertise limits the skills and syntaxes that can be taught, thus hindering the learning of specialized languages such as C, C++, Matlab, Python, Arena, Strata, ArcGIS, among others, which are programming languages that could be useful for students as professionals in many careers.

To alleviate these challenges, professors suggest that high school students should learn skills related to computational thinking and algorithmic logic, so that it would be easier for them to learn and adapt to new programming languages as well as different software during their undergraduate studies. High school students should be exposed to the development of logic, abstraction, decomposition and evaluation, as well as the comprehension of basic algorithmic design, in order to facilitate the learning of programming in higher education.

Introducing students in elementary and higher school to active experiences in algorithmic thinking, abstraction, evaluation, problem solving, decomposition, and debugging will most certainly help develop critical skills from early stages. This may consequently lead to an easier transition from school to university, and would allow professors to focus their undergraduate programming courses on teaching useful features of specific programming languages, as well as reinforcing but not developing CT skills. This idea will be further discussed in the following section.

**Teaching and Learning Programming in Schools**

Barr and Stephenson (2011) argued that today's students would live and work in a world that is heavily influenced by computing principles. In the midst of this trend, CT has become quickly a prerequisite skill for many endeavors of the 21st century. It is a skill that empowers students and thus it is highly beneficial for them to be competent in CT (National Research Council (NRC), 2010). Given that CT is permeating several fields in which most of the students will develop an undergraduate program, it is important to introduce these concepts in schools. Specifically, computational thinking concepts must

appear as early as in elementary school, and continue through secondary school and beyond (Qualls & Sherrel, 2010). This section will discuss the different strategies that have been proposed to teach computational thinking in an effective way (using programming as the central core of computer science), to elementary and high school students.

Deep into the psychology of learning, three theories have been proposed to explain the most effective ways of teaching: the objectivism, the cognitivism, and the constructivism (Luo, 2005). Briefly, the objectivism describes the world as an environment made up of objects, proposing that the learning is based in observation, as a process to create corresponding representations of these objects in the minds of learners (Lakoff, 1987). The cognitivism focuses on how human memory works to promote learning. It considers learning as "involving the acquisition or reorganization of the cognitive structures through which humans process and store information" (Good & Brophy, 1990). Finally, Ben-Ari (2001) stated that the dominant theory of learning in all the areas of knowledge, including computational thinking, is the constructivism. This theory proposes that knowledge is actively constructed by the student using experiences and previously acquired knowledge, and not passively absorbed from textbooks and lectures. Furthermore, the author claims that in terms of CT, passive learning will likely fail because each student brings a different cognitive framework to the classroom, and each student will construct knowledge in a different way. Thus, learning must be active; the student must construct knowledge assisted by guidance from the instructor and feedback from other students.

Based on a constructivist approach, programming has been suggested as the main strategy to address CT in schools (Hromkovic, 2006; Wing, 2006a). Although CT is broader than programming, the latter is a central process for CT. Programming encourages

creativity, logical thinking, precision, and problem–solving, and helps to improve the personal, learning, and thinking skills required in the modern school curriculum (National Research Council (NRC), 2010). Generally, programming is taught using programming languages; nevertheless, teaching programming for elementary and high school students can be extremely difficult. Linn (1985) stated an ideal chain for learning computer programming, which gradually goes from program comprehension and ends with program generation, based on lectures and practical exercises. The chain has three main links: Single Language Features, Design Skills, and General Problem–Solving Skills (Saeli, Perrenet, Jochems, & Zwaneveld, 2011). According to Linn (1985), the ideal chain should start with the understanding of the language features, knowledge that can be assessed by asking students to reformulate or change a language feature in a program so that the program does something different (Saeli et al., 2011). The second link of the chain consists of design skills, which are a group of techniques used to combine language features to form a program. The third link of the chain is useful for learning new formal systems. Asking students to solve problems using an unfamiliar formal system such as a new programming language can help transmit problem–solving skills (Saeli et al., 2011). Although this chain of cognitive accomplishment requires an extensive amount of time, it forms a good summary of what could be meant by deep learning in introductory programming (Robins et al., 2003).

Even if the previously mentioned approach to teach programming has been well described and widely used, introductory courses of programming in schools, called CS0 (Computer Science 0) and CS1 (Computer Science 1) may have as high as a third of the students failing the course when only using lectures and lab–based exercises (Bennedsen &

Caspersen, 2007). A recent study showed that despite the advances in pedagogy, research, and application, students still fail in trying to learn programming, and more important, the course failure rates are not yet, substantially influenced by aspects of external teaching context, such as the programming language taught in the course (Watson, 2014). In other words, the problem might not be the language used to teach programming, but the teaching intervention approach used to reach the attention and interest of the student, as well as the support provided in the process.

**Challenges faced by early programmers, programming languages, and pedagogical tools used in schools.** Over the past 15 years, in most of the schools where programming is taught, programming languages that are professional in nature (such as C, C++, and Java) are used as the main tool for teaching based on the availability of engineers able to teach the course, online material, and third party vendors who provide support around the world (Costa & Aparicio, 2014; Grandell, Peltomaki, Back, & Salaskoski, 2006b). However, these languages have extensive and complex syntaxes, leading to great difficulties for beginners (Jenkins, 2002). The methodology involves traditional teaching methods, normally based on lectures and specific programming language syntaxes, in combination with guides and computer lab sessions (Bennedsen & Caspersen, 2007). Nevertheless, these approaches often fail in motivating students and in getting them involved in deep programming activities (Lahtinen et al., 2005; Schulte & Bennedsen, 2006). One of the difficulties faced by novice programmers include dealing with the verbosity in specialized programming languages, which makes it harder for them to learn algorithmic thinking (Grandell et al., 2006b). Novice programmers have to deal with abstract concepts. They need to understand a given problem, identify the steps to approach such problem and be

able to design a solution. Students, should know how to subdivide a given solution into easy and simpler pseudo–code, and ought to be able to conceive hypothetical error situations in order to test their programs and find possible missteps (Esteves, Fonseca, Morgado, & Martins, 2008). Furthermore, novice programmers have difficulties understanding even the most basic concepts in programming such as variables, data type, machine structure, and functioning, given that there are no real–life analogies related to these concepts (Lahtinen et al., 2005; Miliszewka & Tan, 2007). Last, even after understanding the basic concepts in programming, programmers must learn the specificities of the programming language they intend to use (Lahtinen et al., 2005; Winslow, 1996).

Another important fact that should be considered as a major challenge in learning and teaching programming is the low enrollment and interest of women in computer science courses and programs. Researchers have identified a variety of factors that may contribute to this issue. These factors include concerns about the computing culture, lack of encouragement from peers (Kelleher, Paushc, & Kiesler, 2007), strong lack of motivation, and gender stereotypes (Doube & Lang, 2012). Extremely marked school stereotypes are proven in STEM (Science, Technology, Engineering, and Mathematics) fields including computer science, where females have a lower self–concept of ability than males ranging from early age (Denissen, Zarett, & Eccles, 2007) to postgraduate studies (Cohoon & Asprey, 2006), despite an equivalent and sometimes higher level of achievement (Eccles, 2007). In regards to the STEM field of computing, females have significantly less confidence than males in their ability to learn and succeed (Singh, Allen, Scheckler, & Darlington, 2007). Furthermore, there is a widespread perception that computer experts are socially–isolated “geeks” or “nerds” obsessed with technology, a way

of living that does not match with most women (Margolis & Fischer, 2002) who tend to be more socially oriented (Herring, Christine, Ahuja, & Robinson, 2006). Doube and Lang (2012) showed that when a course related to computing was removed from the School of Science and placed in the School of Humanities, not only was the female enrollment higher but also women admitted to be enjoying the course and finding applicability with their careers. This study concludes that there is a psychological aversion in females due to male stereotyping in computing discipline, which is also observed in mathematics, science, and other disciplines (Lang, 2002). Furthermore, high school women have expressed that their disinterest in computing disciplines arose from their classroom experience, where they faced isolation, low confidence, felt underrated, and extremely anxious (Pau, Hall, & Grace, 2011). To address some of these issues, students from Computer Science and Information Technology, have developed educational videos that enhance learning experiences related to computer programming in a friendly and familiar environment for women (Ali, 2016; Ali, Raza & Ali, 2016.).

Despite the difficulties associated with learning programming in school, which are not easy to solve, it's important to follow Jeannette Wing's vision (Wing, 2006a), in which CT is described as an extensive framework of concepts, abilities, and skills that young people should learn in schools. Serafini (2011) also states that learning programming on an adequate level of abstraction is a very effective didactic approach to CT, independent of the age of the learners. Hence, I decided to gather, classify, and describe below the most powerful and useful tools available to start the immersion of novices in programming.

***Virtual approaches: logo, scratch, and python as alternative languages for beginners.*** Logo is a mini–language, and in contrast to general–purpose programming languages, it

was explicitly developed for teaching programming (Hromkovic, 2006). To write and run programs, novices should not have to deal with the complexity of an extensive general–purpose programming language such as C, C++, and Java (Brusilovsky, Calabrese, & Miller, 1998). Logo editors do not rely on a click–and–drag approach allowing the students to type the instructions by themselves. This makes students care for the correct syntax and allows them to practice debugging (Serafini, 2011). Additionally, the open–source programming environment of XLogo (Le Coq, 2010) presents some interesting features; it runs on multiple platforms, no Internet connection is needed, runs quite well on slow computers and is free, encouraging its use as an introductory tool for programming in schools (Serafini, 2011). Furthermore, teaching materials are available free of charge designed as didactic guidelines and exercises to support teachers (Hromkovic, 2006).

Scratch is a media–rich programming environment developed by MIT's Media lab (https://scratch.mit.edu). It allows students to create animations, games, and interactive art through programming using a computer mouse. The instructions (or code pieces) can be represented as puzzle pieces that only fit one below the other if they are syntactically correct (Malan & Leitner, 2007). Scratch is proposed to be a first language for beginners in introductory courses. It was designed to be user friendly, for all ages, backgrounds, and interests. By using Scratch, students are encouraged to program their own interactive stories, video games, animations, and simulations, and are able to then share their creations with others (Resnick, Maloney, Rusk, & Kafai, 2009). Scratch shows several advantages for introductory programming courses for freshmen students. It allows the instructor and the users to avoid syntax issues (e.g. semicolon use) and thus, allows them to focus on fundamental programmatic constructs (e.g. conditions, loops, variables). Scratch is able to

run as a virtual machine in Linux, Mac OS, UNIX, and Windows. It has a version for kids (called Scratch Junior), and offers a great amount of pedagogical material online for free (Malan & Leitner, 2007). In general, most of the students (76%) that are exposed to Scratch, claim a positive influence on their subsequent experiences with specialized languages such as Java (Resnick et al., 2009).

Python is a high–level scripting language designed by Guido van Rossum to facilitate learning (van Rossum, 1999). Van Rossum suggests that any student could become an expert in programming by using python. The use of Python as a starting programming language, allows novice students to get easily involved into the main characteristics of computational thinking (e.g. solving problems and thinking algorithmically; (Grandell et al., 2006b). This is mainly due to (a) the small, intuitive and clean syntax used in Python compared to languages such as Java or C++; (b) the dynamic typing (variables do not need to be declared) which reduces the notation; (c) the immediate feedback of potential errors; (d) the enforced structural design that leads the programmer to an indented and structured way of writing; (e) and finally, Python is a free and widely used language that comes with a specially designed text editor, tutorials, books, course material, exercises, and assignments available on the Web.

***Real life experiences: LEGO® and LEGO® Mindstorms robot.*** According to Hood and Hood (2005), the use of LEGO® bricks is a powerful method for teaching programming and language concepts. The combination of colors and positions help indicate specific actions that lead to the learning and development of a set of sequenced instructions, algorithmic thinking, and problem–solving strategies, all, critical skills in computational thinking. Moreover, whereas the LEGO® brick approach does not directly involve

technology, it allows the participation of a more inclusive audience given that it evades the intimidation related to the use of computers (LEGO & A/S., 2007). Another advantage of using LEGO® bricks to teach programming, is that a base plate and different–shaped LEGO® blocks are easily acquired. A knowledgeable instructor can design multiple exercises and provide clear instructions by using a single base plate for each participating student or team (Hood & Hood, 2005).

The educational robots of the LEGO Company (LEGO Mindstorms "LM", http://mindstorms.lego.com/) have been broadly used for introducing programming to novice students (Dagdielis, Sartatzemi, & Kagani, 2005; Hussain, Lindh, & Shukur, 2006; LEGO & A/S., 2007). This approach is inspired on the Constructionism learning theory (Papert, 1993), where students are able to construct the knowledge for themselves. Lego Mindstorms are easily programmable robots that contain a broad variety of bricks, sensors, and motors. These robots can be programmed by using proper software environments to execute different kinds of orders, and most importantly, to react to different stimuli received through their sensors (Atmatzidou, Markelis, & Demetriadis, 2008). By using these educational robots, students learn by playing, creating new knowledge based on a pre–existing one (Hussain et al., 2006; LEGO & A/S., 2007).

There are several advantages correlated to the use of LM in introductory programming courses. By using the LM educational robots, students learn through direct observation of their robots operating and interacting in the real world. Students can see when they have made a mistake given that the robot may fall off from a table, crash into a wall, or not interact with environmental conditions. This stimulates them to create robust, expanded, complete, and correct programming, and to invest a huge amount of time

debugging programs (Lawhead, Bland, & Schep, 2003). Furthermore, it has been noted, that the use of LM robots reduces the fear factor of using computers even if the student had had no previous exposure to playing with LEGO® bricks as a child (Lawhead et al., 2003). Huang, Yang, and Cheng (2013) showed that students who used LM to design and build an artifact as a mean to learn basic concepts, performed better in standardized programming tests in comparison to students who learned using standard methods such as flowcharts. Atmatzidou et al. (2008) state that the engagement of children during a course that used LM educational robots greatly contributed to the understanding of programming concepts (such as counter, flag, repetition), leading to the familiarization of structured programming principles.

***Virtual life experiences: game–programming.*** Video games have been widely used in introductory programming courses as an engaging tool to teach computer science, graphics, software engineering, and network topics (Al-Bow, Austin, & Meyer, 2009). Researchers and educators at schools are confident that this tool can be very helpful in the immersion of novices into programming (Rodger, Hayes, Lezing, & Slater, 2009). As Al-Bow et al. (2009) highlighted, the number of papers suggesting the use of video games in introductory programming courses is quite large as it can be further noted in recent proceedings of conferences such as ACM SIGGRAPH Sandbox, Future Play, Conference on Game Development in Computer Science Education, and Foundations of Digital Games.

An example of video game programming is Alice (Conway, Audia, Burnette, & Christiansen, 2000). This educational software uses an innovative programming environment to create 3D animations using drag–and–drop programming as well as a language that is closely related to objects in Java. In this software, each property, method,

and function is attached to an object, with the world being the global object. A variation of Alice, known as Storytelling Alice was created by Caitlin Keller (Kelleher et al., 2007) to encourage middle school students, in particular middle school girls, to learn programming through the creation of short 3D animated movies. Students are able to better understand a variety of programming concepts (e.g. abstraction, modeling, control structures, handlers) by using Alice to make games (Werner et al., 2012) and stories (Kelleher et al., 2007).

A second example of video gaming as an introductory strategy to programming is the Greenfoot environment (www.greenfoot.org). This visual and interactive software has been specially designed to be simple and easy to use for beginners. Greenfoot includes tutorials that provide the necessary Java concepts (such as inheritance, abstraction, and data binding) to create 2D games (Greenfoot team, 2009). In a summer camp organized for 14–and 15–year–old students (Supported by the NSF and Electronic Arts®), the use of computer games such as Greenfoot was used to integrate computer science and art design instruction through a project–based model methodology. This approach, led to a significant improvement in self–confidence and computer programming knowledge (Al-Bow et al., 2009).

A third example of an educational gaming environment for teaching and learning programming is Pex4Fun (from Microsoft Research; http://www.pexforfun.com/). This Web–based tool, allows students to edit code in any browser, to create or manipulate games, and then execute and analyze the game in the cloud (Tillmann & de Halleux, 2011). Pex4Fun is able to find interesting and unexpected input values that help students understand what the code is actually executing. This educational gaming environment, permits a connection between teachers and students by tracking and streaming progress

updates in real time. Pex4Fun has been used to teach and learn software engineering from high school all the way through graduate school courses (Tillmann & de Halleux, 2011).

The last example for video gaming learning is the Hour of code. Code.org® is a non–profit organization launched in 2013, and created in collaboration with engineers from Microsoft, Google, Twitter, and Facebook. This Web page was created with the goal of increasing computer science in schools, as well as encouraging participation of women and underrepresented students of color in this discipline of study (Code.org®, 2013). The strategy is based on learning basic coding principles through gameplay, using an extensive set of tutorials (called the hour of code) containing practice exercises, lectures, and videos designed by technologists including Bill Gates and Mark Zuckerberg, and artwork from popular games like Angry Birds (Rovio) and Plants vs. Zombies (PopCap Game). According to the creators, by now, over ten million people have tried the hour of code (48% female). Among the surveyed teachers who have signed up to teach the intro courses, 99% (almost 115,000) recommend implementing the Web site resources into computer science curricula (Code.org®, 2013). Lastly, one additional advantage is that tutorials can be implemented online, using a smartphone with or without Internet connection, and free of charge.

**Current Status of Teaching Programming in Schools Worldwide.** I have identified three main categories to describe the situation of teaching programming in different countries. These categories are: (a) countries where programming is taught as a core course within computer science based on a national curriculum, (b) countries where programming is taught as a tool related to ICT, but major changes are on the way, and (c) countries where programming is taught as a tool merely related to ICT.

***Countries where programming is taught as a core course within computer science.*** According to Diagiene, Jevsikova, Schulte, Sentance, and Thota (2013), Germany and Israel are the countries where a strong curriculum in computer science (CS) is well established. This curriculum includes problem solving, programming, as well as other skills related to CT. In primary and secondary schools in Germany, students are exposed to robotics and HTML as an introduction to the main topics related to CS in parallel with a strong focus on privacy and ethics. High schools in Germany show a strong emphasis on computer programming and computational theory to develop higher skills related to CS. Israel on the other hand, has developed and implemented a model for a high school CS education program based in four pillars: (a) National CS curriculum and syllabus, (b) Research in CS education, (c) CS teacher preparation programs, and (d) Mandatory CS teaching license (Hazzan, Gal-Ezer, & Blum, 2008). Moreover, students interested in CS take five modules over a two to three year period, which includes: (a) Fundamentals 1 and 2 that introduce central concepts in algorithmic thinking and how to apply them in programming; (b) Software design that introduces data structures, abstract data structures, and designs of complete systems; (c) Second paradigm that introduces object–oriented languages, programming logic, or functional programming; (d) Applications that are focused on computer graphics, management information systems, or Internet programming; and (e) Theory that exposes the students to selected topics in theoretical CS such as models of computation and finite automata. Additionally, it's important to mention that in both countries there is a strong focus on teacher training, ranging from CS bachelor degrees to an extensive pedagogical training certified by the Ministry of Education of each country.

***Countries where programming is taught as a tool for the information and communications technology field, but major changes are on the way.*** There are several countries in which programming has been taught for an extensive period of time (5 to 20 years) as part of a program where students are exposed mainly to the syntaxes of a specific programming language with no clear intention in developing skills related to CT. However, thanks to the efforts of the CS teacher's associations and other individuals related to the computer science field, major changes will become visible in the upcoming future (Jones, 2011). The CS associations in the United Kingdom (http://www.computingatschool.org.uk), the United States (http://www.csta.acm.org), and France (http://www.academie–sciences.fr) have been working on promoting and highlighting the importance of teaching programming as a core course in schools.

The Computing At School (CAS) association in the UK released a document (Jones, Mitchell, & Humphreys, 2013) where they present a review of the CS/ICT curriculum. Several suggestions related to teaching CS, teacher training, and implementation were proposed in view of changing the way how CS is being taught in schools throughout the UK. Moreover, a national curriculum has been proposed in 2013 with the aim of ensuring that students in the UK would be able to understand and apply logic and algorithmic thinking, data representation and communication, as well as analyze and solve problems, and evaluate and apply information technology.

In the USA major events have happened in the past five years. The National Science Foundation (NSF) has launched a program called computing education for the 21st century. Numerous companies and associations including the Association for Computing Machinery (ACM), Microsoft, Google, Facebook, Twitter, and other partners have

developed a coalition called Computing in the Core with the aim of promoting research in CS teaching. Furthermore, they have developed a platform to attract children into programming and CS in general (Stephenson & Wilson, 2012). The Computer Science Teacher Association (CSTA) (2011) has released a revised K–12 CS standards focused on algorithmic and computational thinking concepts.

In France, the French Academy of Sciences released a report in 2013 (French Academy of Sciences (FAS), 2013) claiming that a full review of the ICT/CS curriculum has been made with the support of the government. This ICT/CS curriculum for primary, middle, and high school, aims to achieve a competitive task force in the future. Briefly, students from primary school are dedicated to develop skills related to CT by using appropriate programming languages for their age. This allows them to discover by themselves the main features of algorithmic thinking and problem solving. Later, students from middle school are encouraged to acquire specific concepts related to programming (i.e. syntax) as well as to solve problems on their own, by combining topics in science with programming skills to perform specific tasks. Finally, students in high school work in the strengthening of concepts learned in the earlier two stages, offering the opportunity to introduce advance constructions such as functions, recursion, dynamic allocation, types of data and objects, and parallelism.

In countries such as Finland, Canada, Scotland, Singapore, South Korea, Japan, Greece, and India, there are curricula that include concepts related to CT and ICT. However, the creation of updated drafts of curricula that intend to include learning of CT skills through programming are on the way with the purpose of increasing the number of students in STEM disciplines (Jones, 2011; Sturman & Sizmur, 2011).

***Countries where programming is taught as a tool merely related to the information and communications technology.*** Due to the lack of well–documented information and well–established associations in countries non–mentioned above, it is not unreasonable to assume that most countries are dealing with a situation very similar to that of New Zealand and most countries in Latin America (Jones, 2011). In these countries, computing in school curricula is often ignored, probably due to the fact that computers are used as a tool for teaching (e.g. for web browsing), for general applications related to ICT (e.g. use of Microsoft office), or as a discipline without clear guidelines or purposes regarding CT skills (e.g. learning the syntax of a specific programming language; (Jones, 2011).

In Colombia, there are no clear guidelines provided by the government related to Computational Thinking or Computer Science. Most of the guidelines related to technology focus on the creation of passive users of technology, where students are exposed to a handful of applications with no clear purpose in regards to the development of skills. However, in some private schools, minor changes are occurring. In the last Technology for Education meeting held in Colombia (www.tebinnovation2015.com), besides discussing tools for teaching ICT, some private schools expressed that they have been teaching programming for a while with the purpose of introducing CT skills. Moreover, they spoke about issues related to algorithmic thinking, problem solving, debugging, and other thinking skills. This has motivated them to re–evaluate the curricula and to think about ways in which programming is being taught. A couple of schools expressed their intentions of including CS as an official course within their curricula to develop skills associated to CT. Furthermore, some organizations such as Eduteka (www.eduteka.org) are working on initiatives to expose children to CS and CT. This non–

profit organization, teaches programming via Scratch (Malan & Leitner, 2007) by using mental maps and Venn diagrams. Eduteka also works on strategies to increase the number or girls interested in programming. Another organization from Universidad Nacional de Colombia, developed ProBot (Moreno & Montaño, 2009), a videogame–like application with the purpose of motivating students and engaging them in programming via a boxing–contest environment. Finally, the educational section of RoboCol (an organization in the Department of Engineering of Universidad de los Andes focused on the design and construction of robots for contests robocol.uniandes.edu.co/) teaches programming to kids in elementary schools using the robots of Lego Mindstorms®.

**Current Situation**

Hitherto, approaches to tackle issues associated with teaching and learning programing have been designed to help students in their struggle with programming languages and computational concepts. However, there are not clear strategies to explicitly develop and/or increase skills associated with CT as a first immersion approach. This could be seen as an artifact in traditional teacher-centered curricula, due that teachers and researchers assume that students come with some level development of skills related to CT from previous educational experiences, which should allow students to enter in "transfer processes" rapidly and easily. The severity of this issue rely in that most of the students that reach programming courses could bear a very notorious lack of basic skills, since a large portion of students struggle with math throughout their whole educational life (Bocconi et al., 2016; Buitrago-Florez et al., 2017; Curzon & Mcowan, 2017).

Several studies highlight a strong correlation between performances in math and how students can elaborate in computational thinking. A robust mathematical thinking

comes with high levels of abstraction and algorithmic skills that leads to a rapidly progression in CT through computer programming courses (Bocconi et al., 2016; Buitrago-Florez et al., 2017; Curzon et al., 2014). Nevertheless, traditional teacher-centered education provokes a heavy struggle for many students in mathematical thinking improvement since participation and reification processes are extremely restricted, leaving them with low levels of skills (Skovsmose & Borba, 2004). Therefore, in order to offer an approach that contemplates those aspects, student-centered strategies should be used in initial CT courses to provide an alternative to level-up CT skills, so that students could successfully benefit of this problem solving approach and eventually exploit all the benefits of CT in further programming courses.

# Chapter IV

## Triangular path No 1

Taken and adapted from "Buitrago-Florez, F., Danies, G., Tabima, J., Restrepo, S., & Hernández, C. (2018). Designing a socio-cultural approach for teaching and learning Computational Thinking. *Submmited to the Nordic Journal of Digital Literacy*".

Given the existence of a gap between student-centered strategies and CT development in novice students as explained in the *current situation 1*, this chapter describes the first cycle of critical research subject of this doctoral thesis. First, there is a description of the pedagogical imagination process that leads to the *imagined situation 1,* which is a first version of a socio-cultural CT curriculum design for novice students. Later, following the process of practical organization a pilot course was implemented and qualitatively evaluated as the *arranged situation 1*. Finally, results are analyzed and discussed following the explorative reasoning practice described by Skovsmose and Borba (2004).

### Imagined Situation and Pedagogical Imagination

Programming must be seen as a tool to develop concepts and skills related to CT and computer science (i.e. resourcefulness, problem solving, abstraction, algorithmic thinking, logical reasoning, and debugging), rather than a human–machine communication tool, related to the information and communications technology field "ICT" (French Academy of Sciences (FAS), 2013). Thus, instead of teaching students solely how to write code, CT courses must include within its learning objectives the development of skills related to CT.

Hence, I consider that in order to develop novel CT pedagogical strategies, it is important to clearly distinguish CT from computer programming. Computational Thinking must be seen as the active process of understanding and analyzing a problem, designing a solution to a given problem, and finally implementing the best possible solution for that problem by using different CT skills and concepts. In this journey, a person develops and uses skills related to CT without having to necessarily use a computer at all (Wing & Stanzione, 2016). Computer programming on the other hand, is the process in which CT skills are used to solve problems via a computer. This process takes into account computational concepts such as: variables, vectors, strings, arrays, hashes, and data structures, among several others; and of course, a programming language to interact with the machine.

I want to expand the differentiation between CT and computer programming by using a specific example of the former:

A friend has come up with a wonderful, however complex, cooking recipe and s/he wants to share it with the world. To do so, s/he must write a step-by-step detailed set of instructions on how to prepare this meal. A CT process is the ideal way to help someone in this situation. First, the person must remove all unnecessary details. In this case, it is not necessary to know where to buy the ingredients or the price associated to the recipe (the recipe must be universal, not constrained by the availability of certain resources in a limited area). In order to eliminate these unnecessary details, *abstraction* is used to find all the details that should be avoided. Second, the instructions should be stated in as much details as possible avoiding common sense ideas. *Algorithmic thinking* may then be used to correctly indicate in details the steps for cutting a piece of meat or how to marinate a

specific piece of fish. Additional skills related to CT can be included in a third instance, where *decomposition* has to be used to get a successful recipe for a complex meal. To illustrate *decomposition,* let's assume that the recipe contains three courses: an appetizer, a very complex entrée and a dessert. One of the options available is to think of each course as parallel recipes that must be executed at the same time in order to get the meal served concurrently (nobody wants the main course to be cold, or wants to enjoy the dessert three hours after the meal is done). Computational Thinking also involves *debugging* the steps that are problematic and do not yield the expected results. In this case, going to another friend's house and asking him/her to recreate the recipe exemplify debugging. Under this situation, the original creator of the recipe would observe which steps are missing, which variables might change do to changes in the environment (e.g. a different oven that does not heat enough), or which steps might be unclear. Finally, it turns out that your friend has more than one recipe available to be shared. Here, *generalization* will aid in quickly producing other recipes by using the information and the experience obtained by previous experiences.

In order to avoid confusion among readers, professors and researchers interested in this discussion, one must be aware that CT comprises additional thinking skills, concepts and processes, besides the ones mentioned in this document (Bocconi et al., 2016; Wing & Stanzione, 2016). Nevertheless, the use of basic skills related to CT could be used as an immersion strategy for novices.

According with the conception of critical research, the power of pedagogical imagination lies in identifying the objectives of learning given a current situation, and then establishing coherent activities to guide the learners the successfully achieve such

objectives. This process however, must be grounded in cooperation between teachers and researchers through negotiation and deliberation. As such the imagined situation arises through an extensive dialogue and reflection between me (as teacher and researcher of upcoming CT implementations developed under this project), my advisor, and other teachers interested in this project at Universidad de los Andes.

In this sense, the *imagined situation 1* contemplated a pilot program with a curriculum in which students should be engaged in active experiences related to CT contexts, in order to create a link into their socio-cultural practice. Therefore, the curriculum of the course was divided in two main sections: an initial section (short period of time) in which students will be exposed to specific concepts, definitions and examples of CT, and a later section in which students will be encourage to experience and develop CT skills through Problem Based Learning (PBL) exercises developed in pairs and teams. The pilot program was imagine to be implemented with 10 to 12 volunteer students at Universidad de los Andes, with a time intensity of 2 hours per week during 8 weeks. This time would provide enough qualitative data through written reflections, field dairy of the teacher and a focus group, to perform an evaluation and identify scopes and limitations of the curriculum.

I decide to use a PBL approach to develop skills related to CT due to the large set of benefits that this way of teaching and learning has shown in recent decades. As stated by Capon and Kuhn (2004), PBL strategy displays a superior acquisition of new material, a superior recall of new material and a superior integration of new material with existing knowledge in students. Moreover, in a PBL situation the students use the problem case or scenario to define their own learning objectives (Ribas, 2004) in which CT skills and

concepts could be developed and learned in a most successful way. Additionally, the use of PBL in this specific case would not only tackle CT skills development, but teamwork, sharing of information, independent responsibility for learning, communication skills and respect for others can be approached (Wood, 2003).

Additionally, this first version of a CT curriculum for novice students was designed with a twofold socio-cultural objective: on the one hand, it was critical that students could perform reflection processes throughout the course. On the other hand, the voices of the students and the teacher should be taken into account via narrative instruments for future feedback to the curriculum design. Both would serve as base-ground for a qualitative assessment of the objectives of learning of the course and the establishment of a second current situation for the development of the second iteration of the critical research process.

**Practical organization, Arranged Situation and Results**

Since this is a completely new pedagogical approximation in the development of skills related to Computational Thinking, it is important to execute a preliminary assessment and to report whether CT exercises are appropriate or inappropriate to accomplish the main goal (Teijlingen van, Rennie, Hundley, & Graham, 2001). Thus, the *arranged situation 1* will describe the development and results of the implementation and evaluation of the pilot program under the visions of the *imagined situation 1*.

**Pilot course set up.** To successfully test this novel approach, during the second semester of 2017 a pilot course was developed with 10 student volunteers from Universidad de los Andes. The group of students comprised five female students and five male students ranging from 17 to 22 years, from the undergraduate programs of medicine (2),

microbiology (1), business and administration (1), law (1), design (2), geology (1) and engineering (2). All the students were studying first, second or third semester of their careers. The only parameter used to select the participants was that neither student should have taken courses related to computational programming as part of their coursework.

The students were subsequently engaged in a 16-hours pilot course divided into three main stages. The first stage consisted of a 2 hours lecture, in which the professor explained the concept of CT as well as the definition and characteristics of the five basic skills related to CT (abstraction, algorithmic thinking, decomposition, debugging and generalization). An in-depth explanation of CT and several examples were used in order to create awareness in the students about the differences between CT and computational programming.

Later in the course, two Problem-Based Learning (PBL) exercises were executed in each of the following stages. We decided to use the PBL approach to develop skills related to CT due to the large set of benefits that this way of teaching and learning has shown in recent decades. As stated by Capon and Kuhn (2004), the PBL strategy displays a superior acquisition of new material, a superior recall of new material and a superior integration of new material with existing knowledge in students. Moreover, in a PBL situation the students use the problem case or scenario to define their own learning objectives (Ribas, 2004) in which CT skills and concepts could be developed and learned in a most successful way. Additionally, the use of PBL in this specific case would not only tackle the development of CT skills, but teamwork, sharing of information, independent responsibility for learning, communication skills and respect for others (Wood, 2003).

Therefore, the second stage of the course was a two-moment PBL exercise designed with the objective of encouraging the students to use LEGO bricks in order to perform a controlled and discrete immersion to CT. To successfully carry out the first moment of the exercise, each student was equipped with a standard 720-pieces LEGO bricks box, and was instructed to build a structure of 10 to 15 bricks in companion of a set of instructions (algorithm) so another student may be able to build the same structure by using the same materials later on. Three specific recommendations were given before starting this part of the exercise: *i)* the structure and the algorithm should be elaborated in 90 minutes; *ii)* there were no restrictions for the structure in shape, length, height or any other instruction besides the number of bricks; *iii)* to annotate each step of construction in the algorithm as it was elaborated. Once all the structures and algorithms were made, the students were told to dismantle the structures to subsequently share the algorithm and blocks used with a classmate. Hence, students were arranged in pairs to build the exchanged structure. Students were instructed to annotate and describe the steps in which they had issues while assembling the bricks in the algorithm provided by their partner. To conclude this first part of the activity, students were instructed to debug the algorithm they made given the recommendations of their working pair. This fist step of the PBL exercise was executed in two sessions of 2 hours each.

The second part of the first PBL exercise consisted of instructing the students to modify their algorithms in 60 minutes, so that another student may be able to build an identical structure using the same materials while blindfolded. To do this, the professor shared two recommendations: *i)* to decompose the query structure into simpler structures to be subsequently assembled; and *ii)* to abstract the information that could be useful for a

blind person to build the structure. Once the students ended their modifications, each student was set in pairs to build the structures while blindfolded. One student blindfolded the other, and subsequently read the instructions he/she had modified earlier. Students were instructed to debug in real time if the blinded classmate had issues while assembling the structure. Each pair of students reported each of these issues and modifications. Once the student that was blindfolded finished building the structure, students proceed to exchange roles and build the second structure of the working pair. This second step of this PBL exercise was executed in two sessions of two hours each.

Image 1. Pilot students working on the Lego activity

Finally, the third stage of the course was another PBL exercise that comprised the construction of a 7-step Rube-Golberg Machine. These machines are deliberately complex contraptions in which a series of devices that perform simple tasks are lined together to

produce a domino effect (Rankin, Gooch, & Gooch, 2008a). Hence, students arranged themselves into two groups of three students and one group of four students, and each group was equipped with one construction kit which consisted of strings, springs, wood blocks and sticks of different sizes, metal and wood spheres, pulleys, clay, glue, movement cars, motors, batteries, cables, switches and small light bulbs (for a detailed list of materials see Appendix 1). Students were allowed to use the LEGO brick boxes used in the previous PBL exercise as well. Additionally, three recommendations were given to the groups beforehand: *i)* to explore all the materials given for the activity first; *ii)* to discuss, sketch and construct the final step of the Rube-Golberg machine after material recognition; and *iii)* to discuss and sketch all subsequent steps of the machine previous to its construction. This PBL exercise was executed in three sessions of 2 hours each.

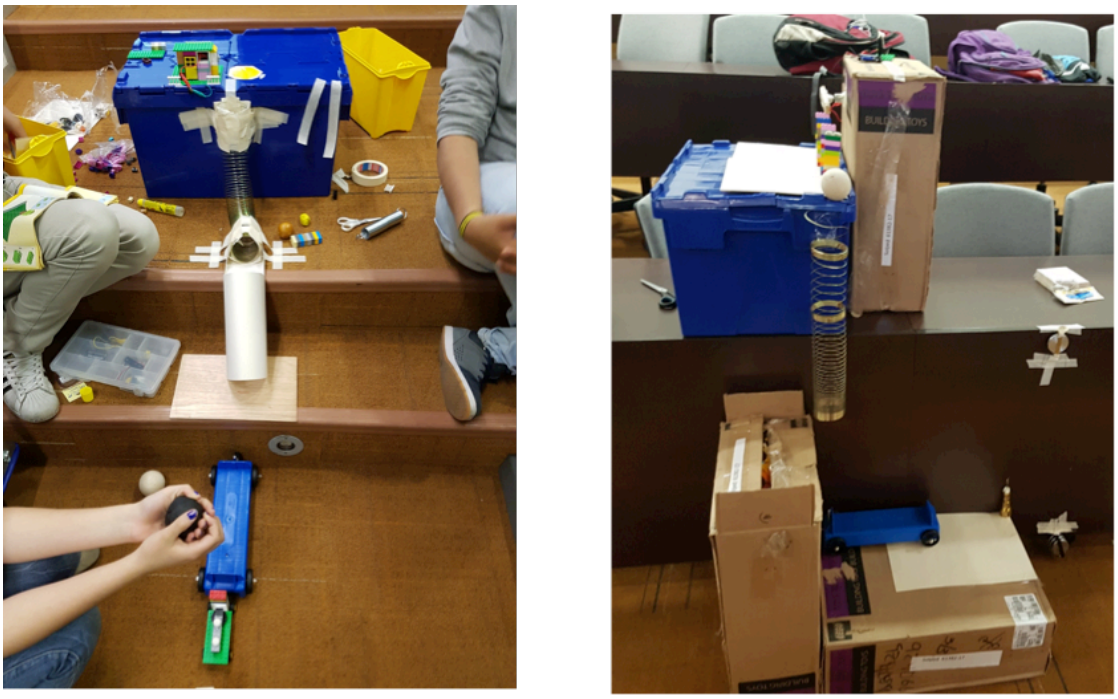


Image 2. Pilot students working on the Rube-Goldberg machine

**Data collection.** A qualitative approach of data collection was used to determine the quality of the methodology in the development of CT skills. A total of three different qualitative instruments were used to maximize the amount of data: A written reflection made by the students at the end of each of the PBL exercises, a diary written by the professor, and a focus group discussion conducted at the end of the pilot course. As such, reflections allow students to explain personal experiences and internal thought activities (Agouridas & Race, 2007). Field diary of teachers offers notes and reflections from another point of view with an extensive amount of extra details than students, who perceive activities and learning processes in a different way (Merriam & Tisdell, 2015). Lastly, focus group provides a dynamic between participants whereby all those present contribute in some way to a deep discussion via a logical sequence of open-ended questions (Lecanda & Garrido, 2003). The written reflections and the focus group were guided with questions that inquired for the comprehension, presence and use of the skills related to CT in the exercises executed. The pilot course as well as the qualitative instruments was developed in Spanish, with a subsequent translation of relevant information to English. For the detailed set of questions included in the reflections and the focus group discussions see Appendix 2. Additionally, students signed an informed consent (previously endorsed by the university ethics committee) in which they authorized the use of anonymized written, visual and audio data for academic purposes.

**Results.** Data from the instruments was organized into five different pre-established categories, which correspond to the five CT basic skills: abstraction, algorithmic thinking, decomposition, debugging and generalization. Additionally, at the end of the pilot course students highlighted a series of aspects they considered important from the pilot course.

This was collected during the focus group discussion and was catalogued into four emerging categories: learning accompaniment and reflection processes, teamwork, communication and creativity.

***Abstraction.*** Students struggled at the beginning with the abstraction concept since it is not very common that professors or others deal with abstraction in a direct way. Curiously, the diary of the professor showed that most of the students thought that abstraction was restricted to art pieces. Nonetheless, students recognized quickly that abstraction is a skill that we use almost every day in an unconscious manner. Furthermore, students were aware that abstraction was a key skill in order to develop the PBL exercises proposed in the pilot course. Some perceptions from reflections regarding the first PBL exercise were:

*St1*: "Abstraction was important to identify the characteristics of the LEGO pieces (shape, color, size) for making the first structure"

*St4*: "I used abstraction to imagine what figure I could make given the time and the pieces"

*St7*: "I think abstraction was important to determine the characteristics of the LEGO bricks that would be useful to create the instructions to assemble the structure while blindfolded"

*St10*: "Abstraction was very useful to imagine, select or discard pieces while I was making my classmate's structure blindfolded"

In the same way, students were able to recognize abstraction in the construction of the Rube-Golberg machine. Some perceptions from the focus group discussions were:

*St8*: "Abstraction, I think, was present all the time (while making the machine) since we had to imagine what to do with all the materials we had available in order to accomplish the objective"

*St2*: "I think that for example, when my group decided to use a spring as channel for a sphere, abstraction was the skill that enabled us to do so. Then, we abstract multiple times to imagine other, less common uses for all the pieces that we had available for building the machine".

***Algorithmic thinking.*** According to the professor's diary, algorithmic thinking was the skill that resulted more familiar to the students. In several cases, students struggled when not enough specific instructions were provided. Interestingly, when the students wrote their own instructions they realized that specific steps were ignored because they thought were logical somehow, however, their classmates did not necessarily consider those steps 'logical'. Some perceptions from the focus group discussion regarding both PBL exercises were:

*St6*: "I think there were two phases of algorithmic thinking in the first PBL exercise, when you are building the structure and when you are writing the steps. Hence, it was a key skill to be able to transmit the information to the classmates".

*St5*: "While building the machine it was very interesting, since we had to imagine the algorithm in our heads first".

*St9*: "Since we had several sessions to assemble the machine, we had to use a lot of algorithmic thinking to remember the steps we made in the previous sessions, otherwise, we had to imagine everything again".

*St1*: "I think something very important was that at the end of the LEGO exercise, the professor was very insistent on the level of detail we had to include in the instructions. So when we worked on the machine, all the members revised measurements, shapes, colors and other details all the time".

***Decomposition.*** The professor's diary showed that as in the case of abstraction, students were not familiarized with the concept of decomposition at the beginning. However, as the course progressed, students realized that decomposition skills were very useful to solve the PBL exercises and to communicate their work to their classmates. Some perceptions from the student's reflections and focus group discussion regarding the PBL exercises were:

*St10*: "Decomposition was useful to fragment my LEGO model into small steps, making it easier to write the algorithm later on".

*St2*: "I think that decomposition was present since we had to make a step-by-step instruction for assembling the LEGO structure while blindfolded".

*St3*: "Since the machine was a step-by-step approximation itself, I think decomposition was very clear in the second exercise"

*St6*: "I think decomposition was very useful in the machine exercise, since we actually decomposed some steps in order to make it easier to re-assemble in posterior sessions".

***Debugging.*** From the professor's perspective, debugging was the most interesting skill after students became familiarized with it. Apparently, students are not aware of using this skill very often in their classes or even in their daily life, making their experiences

successful or unsuccessful, without the opportunity of correcting or improving their mistakes in real time. Nonetheless, once the students realized that debugging was a key element given the structure of the PBL exercises, they started to recognize and value debugging activities. Some perceptions from the student's reflections regarding the first PBL exercise were:

*St8*: "I found it very interesting when I got feedback from my classmate in the first part of the first exercise. I realized that several key information points were missing in the instructions".

*St2*: "I could not believe how many steps I did not write in the algorithm. The information I did not include initially, I had found logical. However, this was not the case for my classmate once he tried to build the structure".

*St6*: "I think debugging was very explicit when I had to change some steps in real time so my classmate could make my structure while blindfolded".

*St7*: "Debugging was very present in the first exercise because we had to make corrections in that specific moment. Once I realized that some instructions were missing or unclear, I had to debug fast".

Likewise, students recognize debugging in the construction of the Rube-Golberg machine. Some perceptions from the focus group discussion were:

*St10*: "I think we debugged each time we tested the machine as a set of steps".

*St2*: "It was very interesting to be able to correct the steps in real time and watch how they worked later. This is not an activity one does frequently in class or any other everyday situation".

*St8*: "Every time we were testing the functioning of the machine, we had to see whether things actually hit or moved other things all the time".

***Generalization.*** At first, students did not realize how generalization could be used in PBL situations. However, after being in contact with the LEGO bricks, students started to generalize different concepts. First they recognized the material, systematized their brainstorming activity, and used less time to develop instructions. This led to an increase in the efficiency and efficacy in their actions. Later, students generalized multiple times when they sketched the steps of the Rube-Golberg machine. Some perceptions from the student's reflections and the focus group discussion regarding both PBL exercises were:

*St7:* "I think generalization was quite present because, for instance, in the LEGO structure I constructed, which looked like a tower, the instructions became a repetition of steps, that seemed like a loop".

*St1*: "Well I consider that once we were in the blindfolded activity, generalization was very useful to perceive, as time passed, the instructions our classmate actually understood".

*St7*: "(in the creation of the Rube-Goldberg machine) we designed some walls using LEGO bricks to generate a chain reaction, this walls were like generalized loops".

*St3*: "In my case I struggled understanding generalization, but when my group used the motion car, we realized the car's displacement was always the same after trying several times. Actually, I remember the displacement was 1.40 meters".

*St5*: "We learned to use one of the springs as a channel from a previous (failed) attempt".

***Learning accompaniment and reflection processes.*** Students claimed that two specific actions helped them increase their CT skills awareness and performance. Students valued that the professor was always reminding them in which specific moments they used CT skills as they solved the PBL exercises. Moreover, students found very useful to write reflections about their processes, which made them realize what actions they carried out, what concepts they learned, and what they needed to improve in the future. Some perceptions regarding this category were:

*St9*: "I think that the reflection activity was extremely helpful since I was able to link all the things I had done in the exercises with concepts we were reviewing in class".

*St10*: "I think it is valuable since it is like an auto-evaluation. It allowed me to reflect if I had accomplished the objectives of the course. For example, in the first exercise I struggled a lot since a decided to build a very complex structure. However, in the second exercise I was focused on accomplishing the steps to build the machine. I realized that in the second exercise I was able to see what the problem was, and thus, was able to solve it".

*St5*: "I think all students really appreciated the accompaniment, as we developed the exercises, the professor was observing frequently and making clear the exact moment in which we used a specific skill".

*St4*: "In fact, I think that the professor's guidance was one of the most powerful strategies in this course".

***Teamwork.*** Students recognized that teamwork was enhanced in the second PBL exercise because it was extremely complex to arrive to an agreement in several actions. Nonetheless, as time passed and the professor helped solve issues, students managed to work as a team in an appropriate way. Some perceptions regarding teamwork were:

*St1*: "I think I improved my teamwork skills since we had to agree on the steps of the machine. Sometimes things did not work as we planned, but after our group discussions we eventually came up with a solution. For instance, we had several issues handling the pulley and the motion car, nonetheless we debugged multiple times with the help of the professor until we were able to come up with a solution".

*St7*: "Let's say that my group was a little bit individualist at the beginning, but as we worked on the machine, all members of my group started brainstorming, discussing and working as a team. Actually, I am a very closed person in terms of teamwork, nevertheless, this time I realized that I needed my classmates to successfully accomplish the objective".

*St5*: "There was something very notorious in my group. At the beginning, one team member did not fit very well, but as time passed that person overcame the issues and was very happy and enthusiastic participating in the exercise. I think that over time I understood the person much better and we were able to interact easily".

*St2*: "In fact, the simple action of brainstorming enhances teamwork, that was a key aspect for accomplishing the exercise in the time provided with the guidance of the professor".

***Communication.*** Students were very emphatic about the development of communication competences throughout the pilot course. They realized that interdisciplinary communication could be challenging, as is communicating ideas in a specific way, avoiding common sense actions. They agreed that this pedagogical approach helps them enhance oral communication skills. Some perceptions regarding communication were:

*St7*: "I think that in the making of the algorithms we had to exercise and improve our communication. One thinks that by expressing actions as we normally do, other people will understand, but that is not the case. As I was saying before, in the case of the blindfolded classmate, it was not possible to say put this piece above the red one, we had to be very specific and avoid a lot the common sense".

*St6*: "I learned that not everyone has the same background. That other people do not think as I do. Then, one must give instructions and stop assuming others are ignorant, one must not suppose anything. Once I realized this it was a breakthrough for me".

*St9*: "I think we naturally assume that everyone has our same background. In the university I have noticed that students outside of my discipline cannot understand me when I talk about specific topics in my field of study. Then, this exercises showed me that I need to be very specific sometimes in order to communicate effectively".

*St1*: "After each activity I realized that we are not detailed-oriented and we assume a lot of things when we try to communicate with others. I found very challenging to communicate in an interdisciplinary group at the beginning. I had to try several strategies to communicate what I imagined in the exercises".

***Creativity.*** Students commented that creativity was required throughout the course in order to solve problems in different ways. To imagine non-common uses of the different materials provided to achieve objectives in the PBL exercises allowed them to be creative in several moments. Perceptions regarding creativity were:

*St10*: "I think creativity was present all the time. At the start of the exercise, we had a lot of materials extremely diverse allowing me to imagine a lot of ideas for a single step of the machine".

*St8*: "I think creativity was present for instance in the moment we needed a ball to fall in a specific point. We looked at all the materials and realized that we could use them in different ways to solve the problem. We worked all the time finding new and innovative solutions".

*St6*: "It is like thinking outside of the box. It is like thinking in all-possible functions a material could have. Besides the creativity we display in the machine, I think I had to be extremely creative designing the algorithm for the blindfolded classmate. For instance I struggled a lot, but in the end, I decided to use coordinates to guide the assembly of the structure. When I reflected about the process, I realized that the use of coordinates was very resourceful and efficient".

**Explorative Reasoning**

The data was analyzed by making a triangulation process (Oliver-Hoyo & Allen, 2006) for seeking a convergence and corroboration of results from the written reflections, the diary of the professor and the focus group discussion. Triangulation of the information shows that throughout the development of the pilot course students became familiar with

CT skills as a solving problem approach. As stated by Vihavainen et al. (2014), the set of basic skills related to CT allows any person to approach a problem, analyze it and propose a solution to the given problem. From the perspective of the students, being aware of the definitions, concepts, and uses related to CT skills, allowed them to effectively solve the PBL exercises provided. Hence, the development of skills through this immersion strategy to CT offers a scenario to develop and reinforce CT skills before any type of computer programming is involved in the process of learning.

Once a good level of CT skills is obtained, professors could use videogames, LEGO Robots, or specialized software and platforms to immerse students into computer programming, as well as the use of advance concepts such as variables, strings, arrays and hashes, to increase critical abilities related to CT (Buitrago-Florez et al., 2017). This would lead to the possibility of teaching formal computer programing via programming languages, aiming to develop high-level computing skills such as sequencing flow and control, automation, efficiency and effectiveness constrains, and iterative and recursive thinking.

Overall, the use of active pedagogical strategies that are centered in the learning process of the students showed development, enhancing and awareness of CT skills in participants of this study. Results from this pedagogical process can be interpreted by two specific socio-cultural learning concepts. First, students were able to share experiences, concepts, situations, expressions and practices in order to solve a set of problems given in the course, allowing them to participate actively in the process. Second, students were able to transform the concepts related to the five CT skills into artifacts at the time they were developing solutions for the PBL exercises. The process of transforming abstract

information into real artifacts, called reification by Wenger (1998), allowed students who are beginners to understand, clarify and raise awareness about what they were learning. Therefore, as proposed by Wenger (1998), when participation and reification processes interact a negotiation of meaning environment is generated, an environment in which students were able to acquired new knowledge and skills (Radford, 2008).

Results revealed how reflection processes can act as a twofold action in negotiation of meaning. On the one hand, students were able to recognize and internalize successful practices and experiences, reinforcing concepts and skills related to CT. On the other hand, students were able to identify unsuccessful actions, allowing them to obtain insights into what may have done wrong in order to avoid now-known pitfalls (Agouridas & Race, 2007). Consequently, by using appropriate guidance, reflective processes can turn into effective forms of participation and reification, because students abstract their role and performance in the learning and use of CT skills and concepts, and transform their thoughts in written words that could be seen as artifacts.

Furthermore, the students recognized the professor's accompaniment as a critical step in creating an active learning atmosphere. As proposed by Northedge (2002), to involve the students in the use of specialized knowledge, artifacts and language, the professor must assume the role of a representative member of a specialized community which generates strategies to involve students in specialized actions. In other words, the professor must provide guidance along the process of learning by becoming a bridge that facilitates the transit of the students from a colloquial to a specialized way of communication.

Moreover, results showed that CT framed in a socio-cultural vision of education as proposed by Vygotsky (1978), allowed students to interact in an environment similar to a community of practice as described by Wenger (1998). By sharing PBL situations and experiences related to the process of CT, students were able to enhance and reflect 21st century competences such as teamwork, communication and creativity. The development of these competences have been discussed as major objectives in higher education, aiming to respond to the demands of complex phenomena such as globalization, technological advances, development of new specializations, need for interdisciplinary work, among several others (Becker, 2006; Galloway, 2007). Therefore, this pedagogical approach not only could be used to involve students in CT as a problem-solving approach, but also to create a scenario for competencies development.

The interaction between students and the research teacher in this first cycle of critical research, provide valuable data regarding the scope and limitations of the first version of the socio-cultural CT curriculum, creating a bridge for the development of the second cycle of investigation as described in the following chapter.

# Chapter V

## Triangular Path 2

Taken and adapted from "Buitrago-Florez, F., Danies, G., Roman, M., Restrepo, S., & Hernández, C. (2018). Boosting 21st Century Competences through Computational Thinking and Student Centered Strategies. *Submmited to the Journal of Life Sciences Education"*.

The purpose of implementing a pilot course for CT skills development was to assure how appropriate was the curriculum design to accomplish such goal. Given the exceptional results not only regarding the development of CT skills but to competencies enhancement, an improved version of the curriculum was designed, implemented and evaluated. This chapter describes a new cycle of critical research following the new CT curriculum version, starting by a *current situation 2* base-grounded in skills and competences development through CT fed by the results of the pilot program. Afterwards, an *imagined situation 2* was developed aiming to foster CT skills and key competences in a CT full time course at university level via pedagogical imagination. Later, the practical organization describes some maneuvers in the implementation of the curriculum given several issues to develop the course at Universidad de los Andes, resulting in an *arranged situation 2* as the implementation of the CT curriculum with senior students from a school in Cajicá, Colombia. Results and evaluation of data from this triangular path given the theoretical educative framework are also discussed in this chapter as part of the *arranged situation 2*.

### Current Situation

Key competences have been a matter of research in the last decade. The changes in the workforce from an industrial model of production, to a rapidly transforming technology-driven economy, demand professionals more prepared in terms of knowledge, skills and experiences. Thus, employers in their personnel selection processes have

increasingly valued "soft" skills such as teamwork, communication and creativity. Additionally, Pellegrino and Hilton (2012) provide evidence that people's competences are determinant for occupations and wages, concluding that young people's social skills affect their job prospects in adulthood. In fact, research in health and well-being has shown that competences such as creativity, risk-taking and effective communication can be a more accurate predictor of performance than IQ scores (Ontario Ministry of Education., 2016).

Several studies regarding *life-learning* and *learning to learn* identify four competences that make a measurable contribution to desirable outcomes in educational achievement, relationships, employment, health and well-being; this applies to all individuals, not only to those in a specific trade, occupation, or walk of life (Ontario Ministry of Education., 2016). These competences are commonly known as the 4Cs, representing Critical thinking, Creativity, Communication and Collaboration (P21 association, 2017). As such, critical thinking is defined as the competence that allows an individual for solving problems, critically evaluate information and arguments, and construct meaningful knowledge that is put into practice. Communication is understood as the mastery of digital, writing and speaking skills in a wide range of audiences. Collaboration is the capacity to team up to work effectively and efficiently, displaying interpersonal and team-related skills through the process. Finally, creativity is described as being able to generate novel ideas as well as showing the necessary skills to put them into practice (Fullan & Langworthy, 2014).

Nevertheless, traditional education based on the teaching and learning of concepts related to specific and traditional disciplines fails even more in the development of such competences. Due that the learning process is based on the "transmission" of knowledge

from the teacher to the student, being the latter a passive actor in the practice, students are not allowed to interact, which ultimately is the best way to develop competences (Ontario Ministry of Education., 2016). Hence, the development of 21st century competences require more complex, dynamic, and high-level schemes of concepts and practices to engage the students in more active roles.

Considering the results form the triangular path earlier performed, a socio-cultural CT curriculum could be one of the strategies with a high potential to develop such competences. Therefore, the demand for improvement for the previously CT curriculum towards the development not only of CT skills, but also key competences defined the *current situation 2*.

**Imagined Situation and Pedagogical Imagination**

Vygotsky (1978) socio-cultural perspective of education claims that learning is a complex problem that is the product of different activities, contexts and socio-cultural factors involving the learner. Therefore, the involvement of the learner in meaningful practices is fully required. The first triangular path of this critical research project described a socio-cultural approach for the development of the CT skills as a social process in which the learner's path is embedded in activities of individuals in a particular context. Along this path, the student enters in a community of practice as described by Wenger (1998), as a member who gets involved in participation and reification progressively becoming an expert (Radford, 2008). Hence, only by systematic, intentional and planned actions carefully designed and guided by the teacher students could engage in true learning experiences.

Taking into account the voices of the students and the experiences I've collected as teacher and researcher from the pilot program, as well as several discussions with my advisor and other teachers involved in the fields of CT and education; an *imagined situation 2 emerged*. The CT curriculum previously implemented should be enhanced to foster CT skills and competencies in a way in which critical resources such as time, content of reflection processes, and number activities and participants will increase. These characteristics not only increase the chances for meaningful interactions in a community of learning to accomplish the objectives of learning, but also provide a greater amount of data to validate learning development.

A total of three activities based on PBL approach were proposed - in addition to the CT conceptual lecture- for the development of the new imagined course: the Lego and Rube-Golberg machine from the previous experience, with an additional group activity contemplating the design and construction of a catapult and a structure to hold a projectile from the former. This activity was design to be developed after the Lego activity, as a way in which students could be immerse explicitly in characteristics of the 4Cs. Therefore, skills related to the competences would be enhanced through the Rube-Golberg machine exercise afterwards.

The time of each activity was proposed under the reference of a CBU course, which stands for a basic course for Universidad de los Andes students. This sort of courses are usually developed with 40 to 60 students in a total of 48 hours class, distributed in 3-hour class per week (45 minutes each one) in a total timeframe of 16 weeks. Consequently, the CT lecture could be developed in 2-hour class, the Lego activity in 12 class hours, the catapult activity in 14 class hours, and the Rube-Goldberg machine in 20 class hours. This

time contemplates, material recognition, establishment of objectives of learning, development of the problem situation and reflection processes for each activity. Moreover, an intensive teacher accompaniment and reflection processes, to guide the development of the course and reflect about the moments in which students display characteristics of CT skills and the 4Cs, would frame these activities.

In terms of the evaluation process, this proposal was imagine it to be evaluated by using a mixed-method data analysis in order to maximize the power of the information collected, as well as to strengthen the overall evaluation. The research design should collect information with a majority of open-ended and a minor quantity of close-ended approaches, in line with the embedded mixed method strategies described by Cresswell (Cresswell, 2009). According to (Tashakkori & Teddle, 2002) this research would be mainly qualitative in nature (QUAL-quan), since the purpose was to understand the experience of the participants of the course supported by the gathering of quantitative data of a specific point of research, in this case, problem-solving performance through CT skills.

**Practical Organization, Arranged Situation and Results**

In the last semester of 2017, several meetings between the team of research (me as teacher and researcher in companion of my advisor) and administrative staff from Universidad de los Andes were carried out in order to present and discuss the results of the pilot program, towards the development of a socio-cultural CT CBU course in the first semester of 2018. For the team of research was clear that a course with this characteristics could benefit students form several perspectives, nevertheless, several issues arise from the administrative side, since was somehow unclear how this course fulfills requirements of

CBU guidelines. On the one hand, the problem was that the course could not belong to the Faculty of Education given internal policies, due that my advisor changed her affiliation from this faculty to the Faculty of Engineering in that very semester. On the other hand, the course could not belong to the Faculty of Engineering since it was classified in the education field, according to CBU regulation.

After using some other unsuccessful resources to implement the course such as trying to encompass the course with the deanship of students, I decided to give up about the idea of implementing the course in Universidad de los Andes. Therefore, by using some previous contacts with schools in Bogotá and surroundings, in the end the course was implemented with senior students from a private school from Cajicá, Colombia. The characteristics of the population and the course, as well as evaluation strategy and results are described as follows.

**Participants in the Research Study.** All the participants in the course were students of 11$^{th}$ grade in a private school in Bogotá, Colombia. Every year, this school implements a standard curriculum in all stages, mostly based on traditional teacher-centered strategies in combination with laboratory practices in some disciplines. A total of 42 native Spanish-speaking students (19 female, 23 male), between 16 and 18 years old participated voluntarily in the course. All the participants manifested their willingness to participate by signing a consent form that had been previously approved by both, the school and the university ethics committees. In the case of underage students, parents were notified and signed the consent form as well. Furthermore, students dedicated 48 class hours (one class hour = 45 minutes), distributed in approximately 10 weeks from late February to early June of 2018, in a time frame that was previously dedicated to prepare students for a national test

(Saber 11) that took place at the beginning of February. The school agreed to provide the facilities and time since they are interested in exploring non-traditional strategies for teaching and learning towards a future school curriculum reform.

**Course Development.** The entire curriculum was aligned with the student-centered Problem Based Learning (PBL) approach, which allows students to engage in a problem case or scenario to define and truly understand the learning objectives of the activities (Capon & Kuhn, 2004). As stated by Wood (2003), the true benefits of PBL rely on how students are able to appropriate de problem situation to increase their knowledge, skills and competences, rather than solving de problem per se. The activities carried out in this course were designed and implemented to integrate the use of the five key skills related to CT in combination with a set of interactions in which students could consciously enhance the skills of critical thinking, creativity, communication and collaboration in problem-based scenarios. Three main activities were designed and implemented as follows:

In the first activity, students attended a lecture in order to be familiarized with CT and competence concepts, as well with the description of the first PBL exercise. Later, students were given a 720- piece-s Lego brick box and instructed to build a 15-20 Lego brick structure. Then, they were told to develop an algorithm, meaning a step by step set of instructions so that another student could build the Lego structure they designed, by using the same set of Lego bricks. Afterwards, students were organized in pairs and started testing the couple's algorithm and to debug it in real time. This implied that if a student found an error in the algorithm, the designer was able to fix right away. Once all the students were able to build the structures, they were instructed to modify the algorithm so that a blindfolded mate could assemble it, and later they did the same process of testing and

debugging in real time. Finally, the students went through a reflection process in which they described the difficulties they had, and how they were able to solve them. They also established a relation between the exercise and the use of CT skills, communication and creativity, which were the learning objectives of the activity. The total time provided for this activity was 12 class hours.


Image 3. Full course students working in the Lego Activity

In the second activity, the students teamed-up in groups of three and were given the following problem situation: they had to design, assemble and test a structure able to hold an impact of 300 Newton's force from a handmade catapult, which they had to built as well. Teams were subsequently equipped with the 720-piece Lego brick box, popsicle sticks, springs, rubber bands, strings, metal balls, cardboard and clay. Later, students brainstormed in their groups about the possible learning objectives of this activity, taking into account CT skills, concepts, and characteristics of the 4 C's. Finally, students fully engaged in the development of the structure. The teacher was in charge of constantly monitoring the activities of the groups and pointing out the exact moments in which the groups displayed critical thinking, creativity, communication and collaboration. At the end of the activity the groups showed their final products to the other students. The total time provided for this activity was 14 class hours.

The third PBL exercise consisted in the construction of a Rube-Goldberg machine, which is a set of deliberately complex contraptions in which a series of devices that perform simple tasks are lined together to produce a domino effect (Rankin et al., 2008a). The students were grouped in teams of 4 and first they dedicated some time to understand the Rube-Goldberg machine, recognize the materials and define the learning objectives as they did in the second exercise. They were given the materials previously used in the activities 1 and 2 and additional supplies as described in the Appendix 1. Later, the students brainstormed on the development of the Rube-Goldberg machine and made a sketch before starting the construction process. Subsequently, they fully engaged in the development of the machine for about 12 class hours and the groups showed their machines to the other students at the end of the activity. The teacher monitored the activity making

students aware of the moments they exhibit the 4Cs and students developed a new reflection process in which they described difficulties, strategies to solve them, and an association between the activities carried out and the 4Cs. The total time provided for this activity was 20 class hours.

Image 4. Full course students working in the Catapult and Rube-Golberg machine.

**Evaluation Design and Data collection.** The purpose of this evaluation strategy was to understand the experience of the participants of the course supported by the gathering of quantitative data of a specific point of research (in this case problem-solving). Hence, the research design used in this study collected qualitative and quantitative information, in line with the embedded mixed method strategies described by Cresswell (2009). On the one hand, written reflections from students, notes from the field diary of the teacher, and data from a focus group allowed a qualitative analysis of perception for all competences. Guide questions for the written reflections and the focus group are available in the Appendix 3 and 4. On the other hand, a pre/post test based on multiple-choice questions implemented to gather information about critical thinking was analyzed quantitatively (Appendix 5). The tests were comprised by CT questions from validated sources such as www.code.org and https://teachinglondoncomputing.org/ to inquire about the use of CT key skills. I decided to use this sort of questions given that these skills are considered fundamental for problem solving, a major characteristic of critical thinking (Curzon & Mcowan, 2017).

The evaluation process of this study follows the guidelines of (Bamberger, 2012), who conceives evaluation as "the systematic collection, analysis, and interpretation of information about human phenomena (commonly, social and educational programs) in order to make judgments about their quality and effectiveness – judgments which are then used for decision- making, accountability, improvement, critique, and social betterment, among other uses" (p. 10). Therefore, the techniques used for data collection in this study provide enough data for a triangulation process, in order to establish valid conclusions from the evaluation method (Oliver-Hoyo & Allen, 2006). Following the evaluative approach described above, all 42 students participated in the pre/post tests and in

two written reflections throughout the course. Additionally, 12 students were randomly selected at the end of the course to provide narrative data in the focus group.

**Results.** The aforementioned research design was used to establish whether the curriculum proposed effectively addressed student-learning outcomes and expected perceptions regarding the 4Cs. This section summarizes the analysis of data collected in relation to the four competences.

***Critical Thinking / Problem Solving.*** The information related to problem solving was classified based on the perception of the students regarding their ability to solve problems in the PBL activities throughout the course, in combination with the pre-test/post-test progression analysis. At the beginning of the course, the students struggled with the use of the five key skills related to CT to formulate solutions, reporting that these were abstract for them. Nevertheless, as students developed the problem-based exercises and reflected about their practice, they recognized the usefulness of the CT skills in problem solving.

Data from reflections demonstrate that students were able to identify specific moments in which they used the skills in particular actions in the three PBL exercises (Table 1). This is consistent with Wenger (1998), proposal that learning is a process resulting from participation and reification in specific communities. As students interacted with their peers and the teacher throughout the course, they were able to get involved in a learning community in which the level of expertise and understanding of CT concepts, language, symbols and artifacts increased over time.

Table 2.

*Categories formed from associations between actions and CT skills in reflection 2. The numbers indicate how many students state a specific relationship between a CT skill and an action made in the course activities.*

| CT Skill | Action |
|---|---|
| Abstraction | • Modifying the instructions of the algorithm so a blindfolded student could be able to assemble the Lego structure (23)<br>• Imagining the instructions for building the Lego structure while blindfolded (14)<br>• Imagining how materials could be used to create the catapult and the Rube-Goldberg machine (18)<br>• Sketching the designs for the Rube-Goldberg machine (12)<br>• Looking for alternatives for issues regarding the assembling of the catapult and the Rube-Goldberg machine (9) |
| Algorithmic thinking | • Making the step by step in the Lego activity (39)<br>• Debugging and re-think steps of assembling process of the catapult and the structure (13)<br>• Designing the steps of the Rube-Goldberg machine (25) |
| Decomposition | • Dividing the assembling of the Lego structure into different groups of actions to facilitate the process while blindfolded (40)<br>• Dividing the assembling of the catapult and the structure into different process in the second PBL exercise (23)<br>• Subdividing the assembling of each of the steps of the Rube-Goldberg machine into different steps (16) |
| Debugging | • Fixing errors in real time while another student assembled the Lego structure (35)<br>• Making an auto-evaluation of the instructions for the Lego structure (6)<br>• Testing the resistance of the structure to the impact of the catapult (21)<br>• Testing each of the steps of the Rube-Goldberg machine (7)<br>• Testing the functioning of the Rube-Goldberg machine as a whole (13) |
| Generalization | • Re-contextualizing instructions from the first algorithm in the second one in the Lego activity (16)<br>• Using ideas from a physics class to build the catapult (18)<br>• Using ideas from the exercise 2 in the exercise 3 (16) |

Additionally, in the focus group the students agreed to the fact that being exposed to a CT lecture at the beginning of the course was interesting. However, the application of the CT concepts and skills explained was difficult to understand. In the terms of (Wenger, 1998), they were unable to reify just by attending and listening. Nevertheless, as students became participants of the learning community they had the opportunity to engage multiple processes that allowed them to experience situations and reify CT skills and concepts. For example, two students recalled:

*St2: "I think the lecture at the beginning was very good. However, I did not understand very well how the skills could be applied in the real world. As I start developing the exercises, I was able to see the skills in practice, which was very useful."*

*St11: "This (course) was very interesting for me because I was able to see clearly how the skills worked in the process of building the catapult and the structure. Additionally, when I did the reflections I was able to think more deeply how the skills were used, and I was very surprised to see that in fact I used them all".*

These quotes also show that students recognized the value of reflecting about the learning process. Being reflective creates a relation between knowledge and reality by internalizing successful practices and experiences, which subsequently contributes in the creation of an identity in the learning community that empowers the student in the learning process (Roth & Lee, 2004).

Moreover, participants were asked in the in second reflection to extrapolate the use of the CT skills to everyday life situations. Students were able to choose one situation from a pool of three to answer: making a cooking recipe, learning to drive and developing a

monograph. Data shows the ability of participants to propose a CT approximation to solve in-context situations, indicating a powerful increase in the levels of problem solving within high levels of reification in students (Table 2). Most students chose the development of a monograph. Since the students developed a monograph as a degree requirement, this choice is consistent with the socio-cultural idea that situations in context allow students to build knowledge effectively (Roth, 2009).

Table 3.

*CT skills use to approach an everyday life situation. The numbers refer to how many students chose a specific situation.*

| **Situation** | **Skill approach** |
|---|---|
| Cooking recipe (3) | • Abstraction to consider the elements can be used and which ones should be avoided, so another cook could be able to gets them or replaces it with ease.<br>• Algorithmic thinking to develop a step-by-step set of instructions for the recipe.<br>• Decomposition to divide the recipe into different activities.<br>• Debugging to test the recipe and find issues.<br>• Generalization to develop new recipes from the original. |
| Learning to drive (4) | • Abstraction to understand the parts and functioning of the vehicle<br>• Algorithmic thinking to create a set of steps for specific actions like parking, starting the vehicle and setting up mirrors.<br>• Decomposition to divide complex situations and solve them while learning.<br>• Generalization to use previous experiences and incorporates them in the driving process. |
| Monograph development (32) | • Abstraction to understand the problem as a whole situation and propose a solution.<br>• Abstraction to formulate the research question.<br>• Abstraction to decide what kind of information is useful and what kind is not.<br>• Abstraction to answer the research question according to the results |

| | |
|---|---|
| | obtained.<br>• Algorithmic thinking to develop a step-by-step approach to develop the research.<br>• Decomposition to divide the work into parts to ease the process.<br>• Decomposition to divide the monograph into sub-topics to increase coherence in the development.<br>• Debugging to check for errors and solve them.<br>• Debugging to develop experiments and get results.<br>• Generalization to use approaches from previous experiences in the school. |

Furthermore, quantitative data shows a notorious increase in pre-test/post-test performance in CT concepts and skills to solve specific questions as visualized in Figure 5. In the pre-test, the average of correct answers over 19 possible marks was 4.19, with a standard deviation of 1.49, a highest score of 7 and a lowest score of 1. In comparison, the post-test results show an average of correct answers of 14.5, with a standard deviation of 3.0, a highest score of 19 and a lowest score of 8. Furthermore, the population shows a normal distribution and the T-test shows a significant difference with a P-value of $1.7\ E^{-25}$. These results were somehow expected, since various entries in the field diary of the teacher mention that during the course the students manifested that most of them had struggled with math throughout their school education. Several studies highlight a strong correlation between performance in math and how students can elaborate in computational thinking. A robust mathematical thinking comes with high levels of abstraction and algorithmic skills that lead to a rapid progression in CT through computer programming courses (Bocconi et al., 2016; Buitrago-Florez et al., 2017; Curzon et al., 2014). Nevertheless, traditional teacher-centered education hinders the students' progress in mathematical thinking since participation and reification processes are extremely restricted, resulting in low performance in math skills (Skovsmose & Borba, 2004). Therefore, the results of this

study confirm the benefits of including student-centered pedagogical strategies for skill enhancement. As described by Hernández et al. (2015) these strategies effectively allow students to actively participate and reify in a learning community

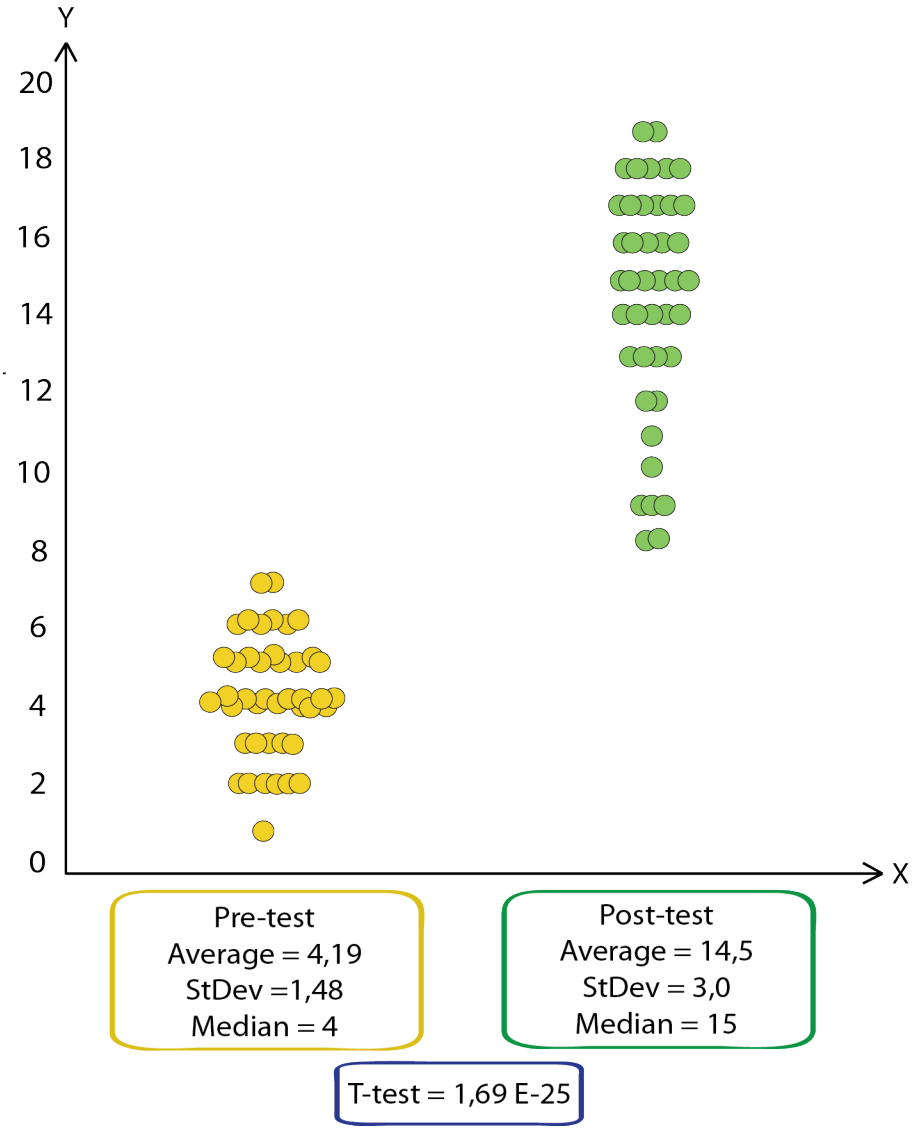



Figure 5. Pre-test and post-test data representation. T-test result shows significant difference in CT skills development throughout the course.

Ultimately, both qualitative and quantitative results show that by the implementation of a CT/student-centered cross-curricular approach, these students were able to understand and use CT skills to solve a wide range of problems. As proposed by the Partnership for 21st century competence development (P21, 2011) students improve their critical thinking competence when they display a good performance in: (1) identifying significant information and concepts to create better solutions, (2) analyzing how parts of a whole interact with each other to produce outcomes in complex systems, and (3) solving different kinds of familiar and non-familiar problems in conventional and innovative ways. Hence, the results show that students not only improved in CT, but the curricular approach was also a major contribution to their critical thinking.

***Communication.*** Analysis from narrative instruments indicate that students recognize that the problem situations they faced throughout the course challenged them in terms of assertive communication. According with entries from the field diary of the teacher, students struggle with complexity of CT language in some situations of dialogue. The teacher expressed however, that students rapidly start to interact between each other by using CT language, expressing situations in terms of abstraction, algorithmic thinking, decomposition, debugging and generalization to be able to listen, transmit and understand ideas from their peers in complex scenarios. This increase in language skills is the result of discussions encouraged by students to clarify, analyze and evaluate their work (Hernández et al., 2015). Some students' perceptions from the focus group regarding these ideas are:

S*t7: "I think we improve a lot over time thanks to being exposed to problem situations all the time, it is not easy to make others understand what one imagine to build,*

*for example the catapult. However, by communicating information in terms of the skills seen in the course, it was easier to understand each other".*

*St10: "... In the exercise of the (Rube-Goldberg) machine we had to discuss several times how to build each of the steps, at the beginning it was not easy to reach common agreements but in the end, I think we learned a lot about how to listen and communicate ideas from/to others".*

Furthermore, students claimed in several opportunities that at the beginning they failed to communicate ideas because they thought some aspects were somehow logical, being the opposite for other students. Nevertheless, as they continue in the process of learning they became reflective and were able to identify these issues and solve them. For example, one student described:

*St7: "...The Lego activities I think helped us to realize that communication it is not easy at all. One take for granted some things that seem logical; nevertheless those are not logical for others. Once I was able to debug this issue, it was easier to transmit instructions to my peers".*

Data from reflection two allowed us to group in six different subcategories moments in which students stated communication was critical to accomplish specific actions: (1) converting ideas into a clear set of steps in the Lego exercise, (2) guiding verbally the blindfolded partner to bypass issues in their algorithms, (3) making understand other students different uses for the materials provided, (4) making understand to others abstract ideas for solving the problems related to exercise two and three, (5) discussing ideas and reach agreements for assembling structures associated to exercises two and three, and (6)

making understand to others issues and debugging processes throughout the course. Perspectives enclosed in these categories are a direct product of being part of a problem-based pedagogical strategy. As proposed by (Capon & Kuhn, 2004), it is not the same to study by listening to the teacher who is in charge to transmit knowledge than a person who has discuss about several topics for an extended period of time. Likewise, the constant communication between all members of the community of learning formed in this course helps students to improve their mastery in CT symbols and language, encouraging effective communication among the students (Wenger, 1998).

***Collaboration.*** Since all the exercises were carefully design to be developed interacting with peers in teams, it was expected to gather valuable information from the narrative instruments about the collaboration competence. From the teacher perspective, students started the course with several issues in effective teamwork, spending a lot of time interacting to take group decisions. However, throughout the progress of the course students successfully developed strategies to take decisions and debug issues regarding teamwork through discussions. Five subcategories regarding specific moments in which students appreciate they had to improve their collaboration practices are: (1) taking into account and integrate all ideas proposed to solve a specific situation, (2) dividing big steps in sub activities with assigned roles with a subsequently debugging process by all members of the teams, (3) appreciating different points of view to find errors in the different exercises, (4) helping other members of the teams to understand problems and encourage them to propose potential solutions, and (5) collaborating in the assembling process of the structures in exercises 2 and 3. This data shows that this problem-based strategy provided scenarios that enabled students to create a community space in which multiple perspectives

were considered and discussed, showing an increasing in collaborative process skills and respect for others (Ribas, 2004).

Students also struggle to collaborate each other in order to understand and facilitate the development of the activities they proposed to tackle the problem situations. Nevertheless, students highlight several times the accompaniment of the teacher as critical, since in multiple occasions he intervened in the groups to facilitate understanding and suggest actions given some characteristics he was able to observe as the students worked in the exercises. As recounted by two students:

*St6: "I did not understand very well the exercise of the catapult at the beginning, however, my teammates and the teacher help me a lot to do it. Once I was sure that I understood, I was able to propose solutions and thus contribute to the team".*

*St12: "For me (collaboration processes) were evident in the development of the (Rube-Goldberg) machine, it was an extensive and very difficult work impossible to do individually, then, we had to request teacher's guidance and increase our tolerance to work as a team to keep going and finish the activity".*

The success of the teacher in the guidance process is a result of previous training in student-centered strategies. Problem-based approaches rely on the capacity of the teacher to understand its role as a supervisor, due that students better resemble a person that aids a less experienced member of the community in the integration of knowledge and actions (Hernández et al., 2015). This role is by all means different from being a transmitter of concepts or a project leader. As Northedge (2002) discusses, a teacher involved in student-

centered strategies must be able to go outside the specialized language and engage students within terms that are familiar for them, until the learner reach mastery in language.

The teacher also had to intervened in all the groups to provide guidance about the difference between teamwork and group work. At the beginning of the second activity teams tried to divide assignments individually, nonetheless, they understood very soon that by implementing that sort of strategies they would expend too much time and would face several issues in the development of the activities as a whole. One student described her experience as:

*St9: "I think this exercises were amazing because here (in the school) we don't usually do that kind of activities as a team, teachers mostly assign tasks in groups, which we divide individually and are joined later. Here we had to unite and think as a team to successfully develop the activities under the times provided".*

This phenomenon can be explained by the enrolment as passive entities that traditional teacher-centered strategies provoke in students (Skovsmose & Borba, 2004). Nonetheless, those sorts of perspectives highlights the way in which, in terms of Johnson, Johnson, and Smith (1991), group accountability and responsibility was reliant on individual accountability and responsibility in this course, being the former a necessary principle for cooperative group learning in communities of learning.

***Creativity.*** Students were very conscious that this competence was base ground in order to developed strategies to solve situations they confronted. Six subcategories were formed given the analysis of students' perception considering the instants when they stated creativity surfaced: (1) imagining different uses of Lego bricks to form a structure in the

first exercise, (2) imagining ways to transmit how to assemble the Lego figure to the blindfolded partner, (3) proposing solutions to issues in real time while the blindfolded partner assembled the Lego structure, (4) imagining non-conventional uses for materials in exercises 2 and 3, (5) proposing, testing and debugging different types of solutions in exercises 2 and 3, (6) understating the objectives of learning facilitate imagining multiple alternative solutions very fast. Consequently, data allow us to conclude that the curriculum design triggers elements of newness, innovation and novelty; inasmuch as problem-solving situations derives in tools and techniques that make the process fun, engaging and collaboration and creates a positive experience that helps the adoptions of new ideas (Awwang & Ishak, 2008). Some opinions product of the interaction in the focus group reinforced this data:

*St11: "I believe creativity was very present. For example, when I did the algorithm in the Lego activity it was very difficult for me, I had to do several attempts and imagine several ways to develop the exercise".*

*St3: "Additionally I think that the unexpected change of instructions for a blindfolded student triggered a lot of creativity. I had to invent several strategies that I think I never would have thought in other class".*

*St9: "At the time of the reflection I realized that in order to create the structure for the catapult impact we had to be creative. It was not easy to develop and debug the structure to hold the projectile".*

*St7: "In the (Rube-Goldberg machine we had to sketch first what we thought were the possible alternatives, nevertheless, once we start to build it we had to change a lot of*

*things to make it work. There were many ways to solve approach the problem that at the beginning we did not imagine".*

Additionally, entries in the filed dairy of the teacher mention two interesting phenomena. On the one hand, students constantly express their surprise as they come up with innovative solutions. In different dialogs with the teacher they claimed that this sort of situations in which they had to be creative, almost does not exist in their everyday curriculum. This idea is an undoubtedly product of traditional education, being the teacher the "transmitter" and the students passive "receptors", thus preventing students to build their own knowledge and depleting creativity as school continues over time (Roth, 2009). On the other hand, students stated that being aware of the objectives of learning and understanding explicitly the reasons why they did each of the exercises, allow them to create solutions and debug more straightforwardly. This is a direct result of interacting with problem-based strategies, in which a central axis of learning is to establish and understand deeply the objectives of learning throughout the process (Ribas, 2009).

# Chapter VI

## Explorative Reasoning

As proposed by (Skovsmose & Borba, 2004), critical research relies in a deep reflection about the development of the triangular paths implemented to elucidate how innovative pedagogical strategies allow teachers and researchers to obtain desire results regarding the objectives of learning. Thus, this chapter discusses and reflects over all the experiences obtained throughout the whole development of this project, in order to answer the research question of this doctoral thesis. As such, this explorative reasoning start by examining the causes that leads to the *current situation 1*, and providing an analysis of educative issues regarding programing learning and teaching. Later, conclusions concerning the innovative pedagogical response established as the first path of critical research are exposed, providing special detail in the differences between the *imagined situation 1* and the *arranged situation 1*. Those differences are analytical due that are the base ground for the *current situation 2*. Subsequently, an analysis of the curricular response to the new *current situation* framed in the second path of critical research is made, to understand the scope and limitations of this doctoral proposal.

Nowadays, it is clear that computation is present in all areas of society, creating an important link between the world's economy, technology, and innovation. The impact of computational skills on multiple fields within the industry, science, communications, humanities, and society in general is becoming more important. Hence, it is crucial to educate our students not only in the passive understanding and application of digital technology, but to teach them the most important principles of how computational aspects

work. In other words, it is extremely important that students should learn the key foundations of CT, given that the skills related to this way of thinking could be very helpful in any career at any stage.

However, in order to achieve these goals, the teaching and learning of such important abilities must be centered and focused around Computational Thinking. The development of skills related to CT such as algorithmic thinking, logic, abstraction, problem solving, debugging, among others, should be the fundamental purpose of programming courses. Moreover, it is important to recall that CT means more than solely learning to program in a specific language, such as Java or C++. Even if programming is central in computing and CT, programming must be taught in light of the development of skills related to it, instead of a passive use of syntax.

Taking into account all the approaches, points of view and reports analyzed in the *current situation 1*, it's clear that there are several issues for novice students participating in introductory courses of programming. Despite the different strategies implemented to support these novice students (ranging from collaborative team works, peer tutors, pre–introductory programming courses, exercises and workshops, virtual platforms, virtual tutors, and forums, to changes in the grading system, programs, and curriculum) multiple adversities are still present when students are confronted to basic CT skills. Furthermore, these difficulties in combination with the exhausting labor of learning syntax, generally leads to frustration, rejection, and poor visions of what programming is, driving most novices away from further programming–related courses.

It is important to mention that there is a critical gap in terms of educational research focused on teaching and learning CT and programming. Despite the significant number of articles written by computer scientists that investigate issues related to teaching and learning in this fields, research conducted by experts from the field of education are almost non–existent in most countries. Because social and cultural factors have a direct impact in learning, they must be taken into account in curricula design. Thus, constructivist, socio–cultural, and other pedagogical approaches are needed to create curricula that are geared towards the development of CT skills in education. These skills will, without a doubt, be fundamental for the vast majority of jobs in the 21st century.

As a pedagogical reaction for the issues regarding skill's development by traditional programming courses, a socio-cultural CT curriculum was designed, implemented and assessed as part of a pilot program. I can summarize the findings of such effort in two main points. First, the curriculum as proposed, could be used as a twofold learning approach by offering a first immersion for freshmen students in CT, as well as a setting for 21st competencies development. Second, students and professors must be aware of the definitions and situations in which each of the CT skills are used. This will allow the students to reflect on their practices and their performance, becoming aware of a powerful tool for CT development. After several discussions around the voices of the students with my advisor and other teachers, I was aware that these ideas must be tested, experimented, and further evaluated. Nonetheless, I was convinced that this novel approach to teach CT could aid on the development of computational skills for a wide range of educational programs, and expand into a new line of research focused in education, formalization, and expansion of techniques for CT and competences development.

Consequently, a socio-cultural CT curriculum for skills and competences development was designed, implemented and tested as the second phase of critical research. This implementation took into account design issues as time, number of activities and types of evaluation, leading to a more complete pedagogical strategy based on qualitative and quantitative approaches. The data supports that by creating and implementing mixed assessment tools and evaluation strategies that facilitate systems to evaluate performance while hearing students' voices is critical in understanding learning experiences. Results from this phase showed that students and the teacher were able to form what I would like to call a "computational thinking community of learning", in which students increase their level of expertise in critical thinking, communication, collaboration and creativity through CT. Evidence showed that most of the students were able to design and implement solutions to the course' problem situations; to associate specific moments in which CT skills were present in their practices; to extrapolate CT benefits to tackle everyday situations; and to increase problem-solving performance quantitatively measured. Therefore, by increasing their ability to solve problem situations students enhanced their critical thinking competence. Furthermore, students displayed and reflect about skills such as listening, dialoguing, clarity, friendliness, open-mindedness, respect, role assignment, failure response, tolerance, curiosity and imagination. Consequently, I can argue that this curriculum proposal creates a learning environment that effectively boosts 21$^{st}$ century competences of great importance for every individual of a knowledge-based society.

Moreover, data analysis demonstrates how reflection processes allow students to build meaningful knowledge. As argued by (Kolmos, Fink, & Krogh, 2004), many researches in the educational field consider that reflection is a key step to perceive and

emphasize the relevance, quality and depth of what is learned in pedagogical environments. Hence, as expressed by (Hernández et al., 2015), being reflective denotes an interaction between the reality and an individual who shapes it according to his/her own practices, thoughts and interpretations. Consequently, in the curriculum design in here exposed the reflection processes became in forms of participation in a specific community of learning, increasing negotiation of meaning.

This phase of the study described the ways in which students were able to form an environment of learning for competences enhancement in a single CT course. As the results are product of an embedded mixed approach, not all the competences were evaluated through quantitative and qualitative instruments, being critical thinking the only competence subject of quantitative analysis. Although students' reflections and perceptions from focus group indicates that participants improved in skills related to the 4Cs, I am fully aware that rubrics for communication, collaboration and creativity progress could be developed in order to quantitatively complement narrative data. Nevertheless, I consider it is important to take into account that perception of students must be the core in assessment of students-centered strategies, as processes of negotiation of meaning are not always easy to quantify.

Furthermore, analysis of narrative data highlights two critical points in which this curriculum design could be improved: teacher assistance and time. Regarding the former, students expressed that more than one teacher or the presence of class monitors could facilitated the process, due that in several cases the teacher could not guide all the questions that emerged in the groups in specific moments. Considering the latter, students claimed that the CT course could last the whole academic year, so they could obtain more benefits

and even apply strategies learned in other school subjects, such as the monograph they had to develop previous the implementation of the CT course. Overall I perceive students' perceptions as appropriate. Formation of large communities of learning often requires more than one expert member to guide processes, as well as more time results in deeply interactions among members. I consider nonetheless, that despite these limitations data collected shows promising results that are encouraging for teachers and administrators interested in student-centered strategies for competences development.

# Chapter VII

The purpose of this chapter is to enclose in specific ideas the findings of this doctoral project to derive conclusions of the work I have made. Later, I would like to explain some lessons I have learned throughout my doctoral formation, which I recognize as critical for the successful development of this project. Finally, some ideas regarding how the results of this doctoral proposal could be exploit it in future research projects, as well as encompassed in the future at Universidad de los Andes (or any other institution) are described.

## Conclusions

According with the socio-cultural vision of education, learning is not considered an individual and isolated process in which a person can adopt a role of "receptor" and somehow absorb knowledge just by listening or repeating specific actions or routines. In fact, this theory of education proposes that knowledge is circulated and transformed among members of a community through their interactions (Wenger, 1998). This means, a classroom could be visualized as a community, specifically as a community of learning, in which the teacher is the most expert member of the community and students displays a diverse range of expertise levels. Understanding a classroom as a community of learning means that several interactions between all members must mediate the path a student recover to increase progressively their level of expertise, enable them to deeply relate with forms of language, sings and artifacts, as well as to create a relationship between knowing and being part of a community (Radford, 2008). Unfortunately, traditional teacher-centered education does not take into account those concepts, making teachers believe that they are

not part of the "classroom" community, but the possessors of a knowledge to be transmitted to the students (Roth, 2009).

Without the systematic implementation of pedagogical strategies to create a sense of community and identity, students enrolled in traditional teacher-centered classes in schools and universities can easily get a feel of insignificance, and may be discouraged to learn. The efforts described in this study combined a focus on CT concepts with personal development, which is very valuable because education systems must pay attention to how students' individual growth relates to social necessities. It is important to recognize that setting specific expectations that connect negotiation of meaning with key skills and competences is defiant, but is a way to respond to the changing demands of a globalized world by preparing students for future challenges. Consequently, I would like to be emphatic in that skills related to CT must be taught to students belonging to any stage of education, in order to foster the cognitive development to form them as strong individuals. In fact, I consider that CT, as in here described, should be recognized and implemented as a discrete subject in schools and universities as is math, biology, or social studies.

Finally, results from this project of investigation offer an extensive scenario to develop and implement further CT curricula, as well as to evaluate it under critical research approach. Teachers could increase the use and difficulty of each of the suggested skills related to CT and the 4Cs by changing the instructions or scenarios of the exercises. Different problem-solving situations will lead to different uses, iterations, and development of CT skills and competencies. Thus, teachers and researchers are encouraged to experiment, randomize the scenarios, and reflect on the products obtained to develop different situations for teaching and learning. Accordingly, student-centered strategies such

as the approach described in this research project could be used as a starting point in schools and universities for CT development, producing an important effect in experience, skills and concepts worth of use to maximize the 4Cs and other competences in later courses in the curriculum.

**Learned Lessons**

Given that my undergraduate (microbiology) and master (biological sciences) programs belongs to fields related to the natural sciences, before I entered to this doctoral program, I thought that qualitative approaches was not powerful enough to provide valid data. Very soon I realized however, that teaching and learning are arenas that fully rely in social exchanges, which must be evaluated by considering the voices and interactions of the stakeholders (students, teachers, researchers and staff). This "illumination" was critical, due that in several cases teachers from engineering, natural sciences and other related disciplines try to evaluate learning as an experiment. This means, those teachers use to see themselves somehow as outsiders from the pedagogical strategies they implement, and try to rely solely on their knowledge and qualitative data from standardized tests to take decisions. Nonetheless, under the socio-cultural vision of education described in this project, that I had the opportunity to explore and experience, it is utterly necessary to establish a dialogue between all stakeholders in order to perform meaningful progression in pedagogical processes.

Furthermore, an important lesson I've learned is that novel ideas in education are extremely difficult to integrate at university level. It was shocking to understand throughout my doctoral program that universities, in particular Universidad de los Andes,

are crystalized institutions very difficult to move forward. There are so many resistances to innovate in pedagogical approaches, that changes rely in a merely form of copying and/or adapting strategies from other countries and institutions, which curiously lead to ignore the voices of our own students. I consider that by overlooking the context and the ways in which students actually learn at institutions, and by avoiding educational research inside the institutions for the institutions, universities and schools would not meet the require level to form students for the upcoming world challenges.

Nonetheless, I've have learned also that research projects like the one described in this document, are vital in the (utterly slow) breaking of the crystallization of learning in institutions. Giving a broad look to the situation, some universities in Colombia (which in turn are not necessary the "biggest" neither the most "important" universities) are putting important value in internal educational research in order to improve their practices. Therefore, I truly think that the findings of this project could benefit higher education in Colombia in the short term. Additionally, taking into account the experience of the second *arranged situation*, I am encouraged to promote these results in schools, so that students could benefit from this approach since early stages of education.

**Future work**

There are interesting ideas for further implementation taking into account the results from this project. As described in the second explorative reasoning for instance, I am well aware that a fully mixed approach for quantitative evaluation competencies development could be elaborated. Nonetheless, as the research question of this study relied in CT skills development, a study with a rubric of such characteristics will be worth of validation in

future projects. Moreover, in the future it could be worth to confront the results in CT-skills development by comparing de performance of students that take this sort of courses and their performance in later programming courses. This way, the benefits of this approach could be somehow more "valid" for policy-makers to take curricular decisions.

Besides, given the benefits that this course could bring to students of any career of any institution, in the long-term this curriculum could be enhanced to create a cross curricular program so all students from an university or an school could develop CT skills and foster the 4Cs. Undoubtedly, this approach could bring more benefits to all students than other traditional teacher-centered cross curricular courses. Also, I'm very interested in verifying how gender concerns in CT/programming/technology could be approached by using strategies like the one in here described, since taking into account what I observed, the underlying characteristics of socio-cultural vision of education *per se* acts against gender issues. Then, a research project around this topic would be interesting to be elaborated.

Finally, I cannot help but think in the benefits this doctoral proposal could bring to Universidad de los Andes. Hence, the most powerful application I can imagine for my institution is to create and validate an entrance CT test for students who have to see programming courses as part of their career program. Thus, students with low scores could have to enroll in a CT course in order to enhance basic skills before entering in programming courses. Actually, I would design a test that could make a rigorous stratification of CT and other programming related skills, so that depending on the score obtained by a student, s/he could take different CT or programming courses. This could

prevent the wasting of already talented students, who probably are able to enter directly to high-level courses to develop more powerful skills related to programming and CT.

## References

Agouridas, V., & Race, P. (2007). Enhancing Knowledge Management in Design Education Through Systematic Reflection Practice. *Concurrent Engineering: Research and Applications, 15*, 63-76. doi: https://doi.org/10.1177/1063293X07076267

Ahmadzadeh, M., Elliman, D., & Higgins, C. (2005). An analysis of patterns of debuggi among novice computer science students. *ITiCSE 2005. ACMPress*, 84-88. doi: 10.1145/1151954.1067472

Al-Bow, M., Austin, D., & Meyer, S. (2009). Using game creation for teaching computer programming to high school students and teachers. *ITiCSE '09 Proceedings of the 14th annual ACM SIGCSE conference on Innovation and Technology in Computer Science Education*, 104-108. doi: 10.1145/1595496.1562913

Atmatzidou, S., Markelis, I., & Demetriadis, S. (2008). The use of LEGO mindstorms in elementary and secondary education: game as a way of triggering learning. *Workshop proceedings of SIMPAR*, 22-30. doi: Retrieved from http://www.terecop.eu/downloads/simbar2008/atmatzidou_et_al.pdf

Awwang, H., & Ishak, R. (2008). Creative Thinking Skill Approach Through Problem-Based Learning: Pedagogy and Practice in the Engineering Classroom *World academy of science, engineering and technology, 16*, 635-640. doi: Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.306.7430&rep=rep1&type=pdf

Bamberger, M. (2012). *Evaluation for equitable development results*: UNICEF Evaluation Office. Retrieved from http://www.clear-la.cide.edu/sites/default/files/Evaluation_for_equitable results_web.pdf

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is involved and what is the role of the computer science education community? *ACM inroads, 2*(1), 48-54. doi: 10.1145/1929887.1929905

Becker, F. (2006). Globalization, curricula reform and the consequences for engineers working in an international company. *European Journal of Engineering Education, 31*, 261-272. doi: https://doi.org/10.1080/03043790600644749

Ben-Ari, M. (2001). Constructivism in computer science education. *Computers in Mathematics and Science Teaching, 20*(1), 45-73. doi: 10.1145/274790.274308

Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming. *SIGCSE Bulletin, 39*(2), 32-36. doi: https://doi.org/10.1145/1272848.1272879

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). Developing computational thinking in compulsory education – Implications for policy and practice. *EUR 28295 EN*. doi: https://doi.org/10.2791/792158

Bruce, K. B. (2005). Controvesry on how to teach CS1: a discussion on the SIGCSE-members mailing list. *ACM SIGCSE Bulletin, 37*(2), 111-117. doi: 10.1145/1041624.1041652

Brusilovsky, P., Calabrese, E., & Miller, P. (1998). Mini-languages: a way to learn programming principles. *Education and Information Technologies., 2*, 65-83. doi: 10.1023/A:1018636507883

Buitrago-Florez, F., Casallas, R., Hernandez, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a Generation's Way of Thinking: Teaching Computational Thinking Through Programming. *Review of Educational Research, 87*(4), 834-860. doi: https://doi.org/10.3102/0034654317710096

Butler, M., & Morgan, M. (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts. *Proceedings ascilite Singapore 2007*. doi: Retrieved from http://www.ascilite.org/conferences/singapore07/procs/butler.pdf

Capon, N., & Kuhn, D. (2004). What's so good about Problem Based Learning? *Cognition and instruction, 22*, 61-79. doi: https://doi.org/10.1207/s1532690Xci2201_3

Chase, J. D., & Okie, E. G. (2000). Combining cooperative learning and peer instruction in introductory computer science. *SIGCSE Bulletin, 32*(1), 372-376.

Code.org®. (2013). Hour of code kicks off to introduce K-12 students to computer programming. *IEEE Computer Society*. doi: Retrieved from https://code.org/educate

Cohoon, J., & Asprey, W. (2006). A critical review of the research on women's participation in postsecondary computing education. *Women and Information Technology: Research on Underrepresentation. The MIT Press.*, 137-180. doi: 10.7551/mitpress/9780262033459.003.0005

Computer Science Teacher Association (CSTA). (2011). Computer science standards. *ACM*. doi: Retrieved from https://csta.acm.org/Curriculum/sub/K12Standards.html

Conway, M., Audia, S., Burnette, T., & Christiansen, K. (2000). Alice: lessons learned from building a 3D system for novices. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 486-493. doi: 10.1145/332040.332481

Costa, C. J., & Aparicio, M. (2014). Evaluating success of a programming learning tool. *Proceedings of the international conference on information systems and desing of communication*, 73-78.

Cresswell, J. (2009). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*: 3rd ed. Thousands Oaks, CA:Sage.

Curzon, P., Dorling, M., Ng, T., Selby, C., & Woollard, J. (2014). Developing computational thinking in the classroom: a framework. *Computing at School (CAS) Releases*. doi: Retrieved from https://eprints.soton.ac.uk/369594/

Curzon, P., & Mcowan, P. (2017). *The power of computational thinking: games, magic and puzzles to help you become a computational thinker*: World Scientific Publishing Europe.

Dagdielis, V., Sartatzemi, M., & Kagani, K. (2005). Teaching with robots in secondary schools: some new and not-so-new pedagogical problems. *ICALT'05 - Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies*. doi: 10.1109/ICALT.2005.255

De La Mora, N., & Reilly, C. F. (2012). The impact of real-world topic labs on student performance on CS1. *Proc. Frontiers in education*, 1-6.

Denissen, J., Zarett, N., & Eccles, J. (2007). I like to do it, I'm able, and I know I am: Longitudinal couplings between domain-specific achievement, self-concept, and interest. *Child Development., 78*(18). doi: Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/17381782

Diagiene, V., Jevsikova, T., Schulte, C., Sentance, S., & Thota, N. (2013). A comparison of current trends within computer science teaching in school in Germany and the UK. *Informatics in Schools: Local Proceedings of the 6th International Conference ISSEP*. doi: Retrieved from https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/6187/file/cid06.pdf#page=65

Doube, W., & Lang, C. (2012). Gender and stereotypes in motivation to study computer programming for careers in multimedia. *Computer Science Education, 22*(1), 63-78. doi: Retrieved from http://eric.ed.gov/?id=EJ958969

Eccles, J. (2007). Where are all the women? Gender differences in participation in physical science and engineering. *In S.J. Ceci & W.M. William (Eds.), Why aren't more women in science?: Top researchers debate the evidence. American Psychological Association.*, 12. doi: 10.1037/11546-016

Esteves, M., Fonseca, B., Morgado, L., & Martins, P. (2008). Contextualization of programming learning: a virtual environment study. *Journal of Virtual Worlds Research*.

Flowers, T., Carver, C., & Jackson, J. (2004). Empowering students and builiding confidence in novice programmers through gauntlet. *Frontiers in Education Conference, 1*, T3H/10-T13H/13. doi: 10.1109/FIE.2004.1408551

French Academy of Sciences (FAS). (2013). Teaching computer science in France, tomorrow can't wait. *Institut De France - Academie des Sciences*. doi: Retrieved from http://www.academie-sciences.fr/pdf/rapport/rads_0513gb.pdf

Fullan, M., & Langworthy, M. (2014). *A rich seam: How new pedagogies find deep learning*: London: Pearson.

Galloway, P. (2007). *The 21st-Century Engineer: A Proposal for Engineering Education Reform*. Reston, Virginia: ASCE Press.

Good, T. L., & Brophy, J. E. (1990). *Educational psychology: A realistic approach* (4th ed.): White Plains, NY: Longman.

Grandell, L., Peltomaki, M., Back, R.-J., & Salaskoski, T. (2006a). Why Complicate Things? Introducing Programming in High School Using Phyton. *Conferences in Research and Practice in Information Technology, 52*. doi: Retrived from https://dl.acm.org/citation.cfm?id=1151880

Grandell, L., Peltomaki, M., Back, R.-J., & Salaskoski, T. (2006b). Why complicate things? introducion programming in high school using phyton. *Conferences in Research and Practice in Information Technology, 52*. doi: Retrieved from http://dl.acm.org/citation.cfm?id=1151880

Green, J., Bouce, A., & Ahn, J. (2015). *A Values-Engaged, Educative Approach for Evaluating Education Programs: A Guidebook for Practice*: University of Illinois publications. United States

Greenfoot team. (2009). Greenfoot environment and API. http://www.greenfoot.org.

Hazzan, O., Gal-Ezer, J., & Blum, L. (2008). A model for high school computer science education: the four key elements that make it! *SIGCSE Bulletin 40*, 281-285. doi: 10.1145/1352135.1352233

Hernández, C., Ravn, O., & Valero, P. (2015). The Aalborg university PO-PBL model from a socio-cultural learning perspective. *Journal of problem based learning in higher education., 3*, 16-35. doi: https://doi.org/10.5278/ojs.jpblhe.v0i0.1206

Herring, S., Christine, O., Ahuja, M., & Robinson, J. (2006). Gender and the culture of computing in applied IT education. *Encyclopedia of Gender and Information Technology.* doi: 10.4018/978-1-59140-815-4.ch074

Hood, C., & Hood, D. (2005). Teaching programming and language concepts using LEGOs®. *ITiCSE '05 Proceedings of the 10th annual SIGCSE conference on Innovation and Technology in Computer Science Education.*, 19-23. doi: 10.1145/1151954.1067454

Hromkovic, J. (2006). Contributing to General Education by Teaching Informatics. *Informatics education - the bridge between using and understanding computers, 4226*, 25-37.

Huang, K., Yang, T., & Cheng, C. (2013). Engineering to see and move: teaching computer programming with flowcharts vs LEGO robots. *iJET, 8*(4). doi: Retrieved from http://online-journals.org/i-jet/article/view/2943

Hussain, S., Lindh, J., & Shukur, G. (2006). The effect of LEGO training on pupils' school performance in mathematics, problem solving ability and attitude: swedish data. *Educational Technology & Society, 9*(3), 182-194. doi: Retrieved from http://eric.ed.gov/?id=EJ836851

Jenkins, T. (2002). On the difficulty of learning to program. *Proceedings of 3rd Annual LTSN_ICS Conference*, 53-58. doi: Retrieved from http://www.ics.ltsn.ac.uk//pub/conf2002/jenkins.html

Johnson, D., Johnson, R., & Smith, K. (1991). *Active Learning: cooperation in the college classroom*: Edina, MN: Interaction Book.

Jones, S. (2011). International Comparisons. *Computing at School (CAS) Releases*.

Jones, S., Mitchell, B., & Humphreys, S. (2013). Computing at School in the UK. *Computing at School (CAS) Releases*.

Kay, D., van der Hoek, A., & Richardson, D. (2005). Informatics: A focus on computer science in context. *SIGCSE Bulletin*. doi: Retrieved from http://www.ics.uci.edu/~andre/papers/C45.pdf

Kelleher, C., Paushc, R., & Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming. *CHI 2007 Proceedings - Programming by and with end-users*. doi: 10.1145/1240624.1240844

Kolmos, A., Fink, F., & Krogh, L. (2004). The Aalborg PBL model. . *Aalborg University Press*. doi: Retrieved from https://http://www.en.aau.dk/about-aau/aalborg-model-problem-based-learning

Lahtinen, E., Mutka, K., & Jarvinen, H. (2005). A study of the dificulties of novice programmers. *Proceedings of the 10th Annual SIGSCE Conference on Innovation and Technology in Computer Science Education (ITICSE 2005)*, 14-18. doi: Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.472.9952

Lakoff, G. (1987). *Women, fire and dangerous things*: University of Chicago Press.

Lang, C. (2002). Tertiary computing course selection: The impact of mathematics anxiety on female decision making. *AJET, 18*, 341-358. doi: 10.14742/ajet.v18i3.1764

Lasserre, P., & Szostak, C. (2011). Effects of team-based learning on a CS1 course. *Proc. ITiCSE*, 133-137.

Lawhead, P., Bland, C., & Schep, M. (2003). A road map for teaching introductory programming using LEGO© mindstorms robots. *ITiCSE-WGR '02 Working group reports from ITiCSE on Innovation and Technology in Computer Science Education*, 191-201. doi: 10.1145/782941.783002

Le Coq, L. (2010). Xlogo. http://xlogo.tuxfamily.org/.

Leach, J., & Scott, P. (2003). Individual and Sociocultural Views of Learning in Science Education. *Science & Education, 12*, 91-113. doi: https://doi.org/10.1023/A:102266551

Lecanda, R., & Garrido, C. (2003). Introducción a la metodología de investigación cualitativa. *Revisa te psicodidáctica, 14*, 5-40. doi: Retrieved from http://www.redalyc.org/articulo.oa?id=17501402

LEGO, & A/S., D. (2007). Study of Educational Impact of the LEGO Dacta Materials. *Retrieved from* http://www.lego.com/education/download/infoescuela.pdf.

Linn, M. C. (1985). Cognitive consequences of programming instruction in classrooms. *Educational Researcher, 14*(14-29). doi: 10.3102/0013189X014005014

Luo, D. (2005). Using constructivism as a teaching model for computer science. *The China Papers*. doi: Retrieved from http://science.uniserve.edu.au/pubs/china/vol5/CP5_itcs_02.pdf

Malan, D., & Leitner, H. (2007). Scratch for budding computer scientists. *SIGCSE '07 Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, 223-227. doi: 10.1145/1227504.1227388

Margolis, J., & Fischer, A. (2002). *Unlocking the clubhouse: Women in computing.*: Cambridge MA: The MIT Press.

Merriam, S. B., & Tisdell, E. J. (2015). *Qualitative Research: A Guide to Design and Implementation*: Wiley.

Miliszewka, I., & Tan, G. (2007). Befriending computer programming: a proposed approach to teaching introductory programming. *Issues in Informing Science & Information Technology, 4*, 277-289. doi: Retrieved from http://proceedings.informingscience.org/InSITE2007/IISITv4p277-289Mili310.pdf

Moreno, J., & Montaño, E. (2009). ProBot: Juego para el aprendizaje de la logica en programación. *Nuevas Ideas en Informatica Educativa., 5*, 1-7. doi: Retrieved from http://www.tise.cl/2009/tise_2009/pdf/1.pdf

Mow, C. (2006). *The effectiveness of cognitive apprenticeship based learning environment (CABLE) in teaching computer programming*: University of South Australia. Australia.

National Research Council (NRC). (2010). Report of a workshop on the scope and nature of computational thinking. *The National Academies Press*. doi: 10.17226/12840

Nikula, U., Gotel, O., & Kasurinen, J. (2011). A motivation guided holistic rehabilitation of the first programming course. *ACM Transactions on Computing Education (TOCE), 11*(4), 24-38. doi: 10.1145/2048931.2048935

Northedge, A. (2002). *Organizing excursions into specialist discourse communities: A sociocultural account of university teaching.*: Wiley Online Library.

Oliver-Hoyo, M., & Allen, D. (2006). The Use of Triangulation Methods in Qualitative Educational Research. *Journal of College Science Teaching, 35*, 42-47. doi: Retrieved from http://www.nsta.org/publications/news/story.aspx?id=51319

Ontario Ministry of Education. (2016). *21st century competencies: Foundation document for discussion, phase 1: Towards defining 21st century competencies for ontario*. Ministry of Education of Canada.

P21. (2011). Framework for 21st century learning. *Partnership for 21st competencies development*.

P21 association. (2017). Partnership for 21st century learning. 2017

Pane, J., & Myers, B. (1996). Usability issues un the design of novice programming system. *Carnegie Mellon University, School of Computer Science Technical Report CMU-CS-96-132*. doi: Retrieved from http://www.cs.cmu.edu/~pane/cmu-cs-96-132.html

Papert, S. (1993). Mindstorms: Children, computers, and powerful ideas. *Basic Books, Inc. Publishers, New York.*

Pau, R., Hall, W., & Grace, M. (2011). "It's boring": female students' experience of studying ICT and computing. *SSR, 92*(341), 89-94. doi: Retrieved from http://eric.ed.gov/?id=EJ943938

Pellegrino, J. W., & Hilton, M. L. (2012). *Education for life and work: Developing transferable knowledge and skills in the 21st century*. National Research Council. Committee on Defining Deeper Learning and 21st Century Skills, Board on Testing and Assessment and Board on Science Education, Division of Behavioral and Social Sciences and Education. Washington, DC: The National Academies Press.

Piteira, M., & Costa, C. (2012). Computer programming and novices programmers. *ISDOC '12 Proceedings of the Workshop on Information Systems and Design of Communication*, 51-53. doi: 10.1145/2311917.2311927

Piteira, M., & Costa, C. (2013). Learning computer programming. *ISDOC '13 Proceedings of the 2013 International Conference on Information Systems and Design of Communication*, 75-80. doi: 10.1145/2503859.2503871

Qualls, J., & Sherrel, B. (2010). Why computational thinking should be integrated into the curriculum. *Journal of computer science colleges, 25*, 66-71. doi: Retrieved from https://dl.acm.org/citation.cfm?id=1747148&dl=ACM&coll=DL

Radford, L. (1997). On Psychology, Historical Epistemology, and the teaching of Mathematics: Towards a socio-cultural History of Mathematics. *International journal of Mathematics Education, 17*, 26-33. doi: Retrieved from http://www.jstor.org/stable/40248219

Radford, L. (2008). *The ethics of being and knowing: towards a cultural theory of learning*: Sense Publishers.

Rankin, Y., Gooch, A., & Gooch, B. (2008a). The impact of game design on students' interest in CS. *GDCSE conference 08*, 59-63. doi: https://doi.org/10.1145/1463673.1463680

Rankin, Y., Gooch, A., & Gooch, B. (2008b). The impact of game design on students' interest in CS. *GDCSE '08 Proceedings of the 3rd international conference on Game development in computer science education*, 59-63. doi: 10.1145/1463673.1463680

Resnick, M., Maloney, J., Rusk, N., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM - Scratch Programming for All CACM Homepage archive, 52 (11)*, 60-67. doi: 10.1145/1592761.1592779

Ribas, A. (2004). Las líneas maestras del aprendizaje por problemas. *Revista interuniversitaria de formación de profesorado, 24*, 79-96. doi: Retrieved from http://www.redalyc.org/pdf/274/27418106.pdf

Ribas, A. (2009). *Aprendizaje basado en problemas en la educación superior*. Colombia: Editorial Universidad de Medellín.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education, 13*(2), 137-172. doi: Retrieved from http://home.cc.gatech.edu/csed/uploads/2/robins03.pdf

Rodger, S., Hayes, J., Lezing, G., & Slater, D. (2009). Engaging middle school teachers and students with Alice in a diverse set of subjects. *Proceedings of the 40th ACM Technical Symposium on Computer Science Education (SIGCSE'09)*, 271-275. doi: 10.1145/1508865.1508967

Roth, W. M. (2009). *The Gap Between University and the Workplace. Examples from Graphing in Science. In O. Skovsmose, P. Valero, & O. Ravn (Eds.), University Science and Mathematics Education in Transition (pp. 133–155)*: Springer.

Roth, W. M., & Lee, S. (2004). Science education as/for participation in the community. *Science Education, 88*, 263-291.

Saeli, M., Perrenet, J., Jochems, W. M., & Zwaneveld, B. (2011). Teaching programming in secondary school: a pedagogical content knowledge perspective. *Informatics in Education, 10*(1), 73-88. doi: Retrieved from https://http://www.fing.edu.uy/grupos/nifcc/material/2015/PCK.pdf

Schulte, C., & Bennedsen, J. (2006). What do teachers teach in introductory programming? *Proceedings of the Second International Workshop on Computing Education Research*, 17-18. doi: 10.1145/1151588.1151593

Serafini, G. (2011). Teaching programming at primary schools: visions, experiences and long-term research prospects. *5th International Conference on Informatics in Schools: Situation, Evolution and Perspectives, ISSEP 2011*, 143-154. doi: 10.1007/978-3-642-24722-4_13

Shaffer, S., & Rosson, M. (2013). Increasing student success by modifying course delivery based on student submission data. *ACM inroads, 4*(4), 81-86.

Singh, K., Allen, K., Scheckler, R., & Darlington, L. (2007). Women in computer-related majors: A critical synthesis of research and theory from 1994 to 2005. *Review of Educational Research., 77*(34). doi: 10.3102/0034654307309919

Skovsmose, O., & Borba, M. (2004). *Research Methodology and Critical Mathematics Education. In*

*Valero, P & Zevenbergen, R (eds.), Researching the Socio-Political Dimensions of Mathematics Education (vol. 35, pp. 207-226), Kluwer Academic Publishers, Boston (doi:10.1007/1-4020-7914-1_17)*. Dordrecht: Kluwer.

Stake, R. (2004). *Standards-based and responsive evaluation*: Sage publications. United States.

Stephenson, C., & Wilson, C. (2012). Reforming K-12 computer science education... what will your story be? *ACM inroads, 3*(2). doi: 10.1145/2189835.2189850

Sturman, L., & Sizmur, J. (2011). International comparison of computing in schools. *Slough: NFER*. doi: Retrieved from https://http://www.nfer.ac.uk/publications/cis101/cis101_home.cfm

Sutherland, L., Scanlon, L., & Sperring, A. (2005). New directions in preparing professionals: examining issues in engaging students in communities of practice through a school–university partnership. *Teahing and Teacher Education, 21*, 79-92. doi: https://doi.org/10.1016/j.tate.2004.11.007

Syslo, M. (2014). From Algorithmic to Computational Thinking: On the Way for Computing for all Students. *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, 1-4. doi: https://doi.org/10.1145/2729094.2742582

Tashakkori, A., & Teddle, C. (2002). *Handbook of mixed methods in social and behavioral research*: Thousand Oaks, CA: Sage.

Teijlingen van, E., Rennie, A., Hundley, V., & Graham, W. (2001). The importance of conducting and reporting pilot studies: the example of the Scottish Births Survey,. *Journal of Advanced Nursing, 34*, 289-295. doi: Retrieved from https://http://www.ncbi.nlm.nih.gov/pubmed/11328433

Tew, A., Fowler, C., & Guzdial, M. (2005). Tracking an innovation in introductory CS education from a research university to a two-year college. *SIGCSE '05 Proceedings of the 36th SIGCSE technical symposium on Computer science education*, 416-420. doi: 10.1145/1047344.1047481

Tillmann, N., & de Halleux, J. (2011). Pex4Fun: teaching and learning computer science via social gaming. *Software Engineering Education and Training (CSEE&T), 2011 24th IEEE-CS Conference*, 546-548. doi: Retrieved from http://research.microsoft.com/apps/pubs/default.aspx?id=164417

Turns, J., Sattler, B., Yashuhara, K., & Borgford-Parnell, J. (2014). *Integrating reflection into negineering education.* Paper presented at the 121st ASEE Anual Conference & Exposition, Indianapilis, IN.

Tuugalei, I., & Mow, C. (2012). Analyses of student programming errors in java programming courses. *Journal of Emerging Trends in Computing and Information Sciences, 3*(5). doi: Retrieved from http://www.cisjournal.org/journalofcomputing/archive/vol3no5/vol3no5_11.pdf

van Rossum, G. M. (1999). Computer programming for everybody. *Corporation of National Research Initiatives*. doi: Retrieved from https://http://www.python.org/doc/essays/cp4e/

Vega, C., Jimenez, C., & Villalobos, J. (2013). A scalable and incremental project-based learning approach for CS1/CS2 courses. *Education and Information Technologies., 18*(2), 309-329. doi: 10.1007/s10639-012-9242-8

Vihavainen, A., Airaksinen, J., & Watson, C. (2014). A systematic review of approaches for teaching introductory programming and their influence on success. *Proceedings of the tenth annual conference on international computing education research*, 19-26.

Villalobos, J., Calderon, N., & Jimenez, C. (2009). CUPI2 Community: promoting a networking culture to support teaching programming. *International Conference on Computer Supported Education (CSEDU)*. doi: Retrieved from http://cupi2.uniandes.edu.co/sitio/index.php/el-proyecto-topmenu/unicopublicaciones

Villalobos, J., & Casallas, R. (2006). Teaching/learning a first objetc-oriented programming course outside the CS curriculum. *Tenth workshop on Pedagogies and Tools for the Teaching and Learning of Objetc Oriented Concepts.* doi: Retrieved from http://cupi2.uniandes.edu.co/sitio/index.php/el-proyecto-topmenu/unicopublicaciones

Villalobos, J., Casallas, R., & Marcos, K. (2005). El reto de diseñar un primer curso de programación de computadores. *XIII Congreso Iberoamericano de Educación Superior en Computacion, Cali, Colombia*. doi: Retrieved from http://cupi2.uniandes.edu.co/sitio/index.php/el-proyecto-topmenu/unicopublicaciones

Vygotsky, L. (1978). *Mind in society*: Cambridge, Ma.: Harvard University Press.

Watson, C. (2014). Collaborative learning: a case study for CS1 at grinnell college and austin. *SIGCSE '97 Proceedings of the twenty-eighth SIGCSE Technical Symposium on Computer Science Education, 29*, 209-213. doi: 10.1145/268084.268164

Wenger, E. (1998). *Communities of practice: Learning meaning and identity*: Cambrigde university press.

Werner, L., Campe, S., & Denner, J. (2012). Children learning computer science concepts via Alice game-programming. *SIGCSE '12 Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 427-432 doi: 10.1145/2157136.2157263

Wieman, C. (2007). Why not try a scientific approach to science education. *39*, 5-15. doi: Retrieved from https://http://www.tandfonline.com/doi/abs/10.3200/CHNG.39.5.9-15

Williams, L., McDowell, C., Nagappan, N., Fernald, J., & Werner, L. (2003). Building pair programming knowledge through a family of experiments. *ISESE '03 Proceedings of the 2003 International Symposium on Empirical Software Engineering*, 143-152. doi: Retrieved from http://dl.acm.org/citation.cfm?id=943642

Wing, J. (2006a). Computational thinking. *Communications of the ACM, 49*(3), 33-35. doi: 10.1145/1118178.1118215

Wing, J. (2006b). Computational thinking. *Commun. ACM, 49*(3), 33-35. doi: https://doi.org/10.1145/1118178.1118215

Wing, J., & Stanzione, D. (2016). Center for computational thinking. *Communications of the ACM, 59*, 10-11. doi: https://doi.org/10.1145/2933410

Winslow, L. (1996). Programming pedagogy—a psychological overview. *SIGCSE Bulletin, 28*, 17-22.

Wood, D. (2003). ABC of learning and teaching in medicine Problem Based Learning. *BMJ, 326*, 328-330. doi: Retrieved from https://http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1125189/

## Appendixes

**Appendix 1.** List of materials for the development of PBL exercises. The list represents a construction kit each of the teams had as equipment to work.

1 Lego brick standard box with 720 pieces

1 blue container of 120 L for all materials

1 set of 10 weights ranging from 10g to 100g

1 single pulley

1 double pulley

1 triple pulley

2 packs of clay

6 wooden sticks

1 set of 5 springs (5, 10, 15, 20, 30 cm)

1 magic spring

1 metallic 5 cm sphere

1 wooden 10 cm sphere

1 scissors

1 scalpel

3 markers

1 ruler

10 standard marbles

2 Motion cars with auto propulsion

1 measuring tape

1 chronometer

2 nylon ropes of 100 cm

5 wooden cubes of 5 cm

3 wooden cubes of 10 cm

3 cardboard of 50 cm wide

1 standard tape

1 duck tape

1 paper tape

1 10V motor

4 jumper cables

1 light bulb of 10V

1 switch

1 copper cable of 30 cm

2 10V Batteries

3 plastic containers (50, 100 and 150 ml)

**Appendix 2.** Guide questions for reflections 1, 2, and focus group for the pilot program.

*Reflection 1*

1. Describe how did you experience the Lego activity
2. Describe the difficulties you encounter to solve the activity
3. In which moments do you think the 5 skills related to CT were useful to solve the activity (Abstraction, algorithmic thinking, decomposition, debugging and generalization)?

*Reflection 2*

1. Describe what you understand for Computational Thinking
2. Describe three differences between CT and computational programming
3. Describe the five skills related to CT and explain how did you use it to solve the Rube-Golberg machine activity.
4. Given the five CT skills, provide and example in which each one could be used to approach to an every day situation.

*Focus group*

1. Do you think that the theoretical explanation that was made at the beginning of the course is clear?
2. Do you consider that the theoretical and practical difference between CT and computer programming is clear?
3. Do you consider that the exercise performed with LEGO bricks clearly addresses the 5 basic skills of computational thinking? (Abstraction, algorithmic thinking,

decomposition, debugging, and generalization), Why?

4. Do you consider that the exercise carried out with Rube-Golberg machines clearly addresses the 5 basic skills of computational thinking? (Abstraction, algorithmic thinking, decomposition, debugging, and generalization), Why?
5. Do you think the reflection process was useful in this course?
6. Do you think that this course helps the development or increase of the ability to solve problems? What about Teamwork, Creativity and Communication?
7. Please indicate each one of you what think is good or was innovative in this course.
8. Please, each one of you indicates something that you think could be improved in this course.
9. If this course is completed, that is, it lasts 16 weeks, each week with 2 classes of 1:30 minutes. Do you consider that it could have a positive effect on the development of skills that could be useful in the academic and / or personal field for students of the Universidad de los Andes? Would you take the course? Would you recommend it to your friends / classmates?

**Appendix 3.** Guide questions for reflections 1, 2, and focus group for the full course.

*Reflection 1*

1. Read carefully the definitions of each of the skills related to computer thinking provided.
2. Reflect and mention 1 moment in which you consider that each of the skills was used to solve one of the 2 problems provided. (Remember, you assembled a figure and made a manual for someone else to do it, as well as making a modification of that exercise so that a person who cannot see it will arm it).
3. Reflect and mention 2 things that you already knew and reviewed in the classes, 2 things you learned and 2 doubts that you have left of the process.

*Reflection 2*

1. Read carefully the definitions of each of the skills related to computer thinking.
2. Reflect and mention 1 moment in which you consider that each of the skills was used to solve the catapult and the Rube-Golberg machine activities.
3. Analyze and describe how each of the skills related to computational thinking could contribute to a problem of daily life. You can use any of the following examples, however, it is not restricted and you can also use your own example.
    a. Make a recipe for cooking.
    b. Learn to drive.
    c. Make a monograph or extensive research work.
4. Mention at least 3 specific moments in which you consider that the use of communication, collaboration and creativity skills was present throughout the

course.

*Focus group*

1. Do you consider that the exercise carried out with LEGO bricks clearly addresses the 5 basic skills of computational thinking? (Abstraction, algorithmic thinking, decomposition, debugging, and generalization), Why?
2. Do you consider that the exercise carried out with Rube-Golberg machines clearly addresses the 5 basic skills of computational thinking? (Abstraction, algorithmic thinking, decomposition, debugging, and generalization), Why?
3. Do you think the reflection process is useful in this course?
4. Do you think that this course helps the development or increase of the ability to solve problems?
5. Do you think that this course helps the development or increase of the capacity of Teamwork?
6. Do you think that this course helps the development or increase of the capacity of Creativity?
7. Do you think that this course helps the development or increase of the ability to communicate effectively?
8. Please each indicates something that you think is good, stands out or that you thought was innovative in this course.
9. Please, each one indicates something that you think could be improved from this course.
10. If this course was part of your school curriculum, that is, it had a full duration of 1 year. Do you consider that it could have a positive effect on the development of

skills that could be useful in the academic and / or personal field for college students? Would you like to take the course? Would you recommend it to your friends / classmates?

**Appendix 4.** Pre-test and post-test applied in the full course.

*Pre-test*

Instructions:

- Questions 1 and 2 must be answered in the space provided in the end of the test
- Please use pencil for your answers (any colour)

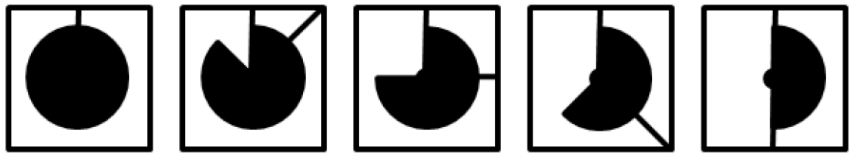
1. What do you know about Computational Thinking?
2. Describe and include one example of what you understand about the following skills:

   Abstraction, algorithmic thinking, decomposition, debugging and generalization

3. In the following sequence

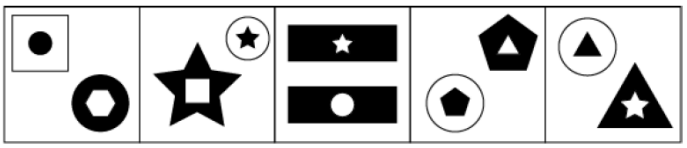

Which figure completes the series?

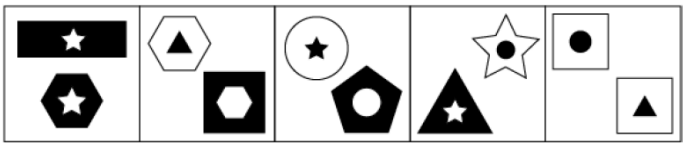
4. In the following sequence



Which figure completes the series?

5. Which set does the Figure belongs to?

Set A 

Set B 

Figure 

6. On a distant planet, the dominant carnivore, the zab, is nearing extinction. The number of zabs born in any year is one more than the (positive) difference between the number born in the previous year and the number born in the year before that.

   Example If 7 zabs were born last year and 5 the year before, 3 would be born this year

   Example 2: If 7 zabs were born last year and 10 the year before, 4 would be born this year.

   2 zabs were born in the year 2000 and 9 zabs were born in 2001. Predict the number of zabs born in 1999 and 2003.

   **Answer**:

7. A word search uses the following special symbols:

   ? Represents a single letter
   * Represents any number of letters, including no letters.

In order for a search term to match a word, it must represent the entire word from start to finish. For example, b * t matches bat but not bath.

How many of the following words does b??st*ing match?

blasting, blustering, boasting, boosting, bootstrapping, bowstrings, bristling, busting, boostings,

(A) 2 (B) 4 (C) 5 (D) 6 (E) 7

*Post-test*

Instructions:

- Use the function "highlight" in yellow to point out your answers

1. Which of the following characteristics belongs to computational thinking?

    a. It is the way in which computers interpret code.
    b. It is a way to approach and solve problems
    c. Can be used in any situation, not only in computational situations
    d. It is algorithmic thinking

2. Which of the following situations represents the use of abstraction?

    a. Making a step by step set of instructions to solve a situation
    b. Looking for errors in real time.
    c. Using a different way of thinking to approach situations, in which unnecessary details are hidden.
    d. Using previous solutions to solve new problem situations.
    e. Dividing a complex problem in little problems to solve the situation efficiently.

3. Which of the following situations represents the use of algorithmic thinking?

    a. Making a step by step set of instructions to solve a situation
    b. Looking for errors in real time.
    c. Using a different way of thinking to approach situations, in which unnecessary details are hidden.
    d. Using previous solutions to solve new problem situations.
    e. Dividing a complex problem in little problems to solve the situation efficiently.

4. Which of the following situations represents the use of decomposition?

    a. Making a step by step set of instructions to solve a situation
    b. Looking for errors in real time.
    c. Using a different way of thinking to approach situations, in which unnecessary details are hidden.
    d. Using previous solutions to solve new problem situations.
    e. Dividing a complex problem in little problems to solve the situation efficiently.

5. Which of the following situations represents the use of debugging?

    a. Making a step by step set of instructions to solve a situation

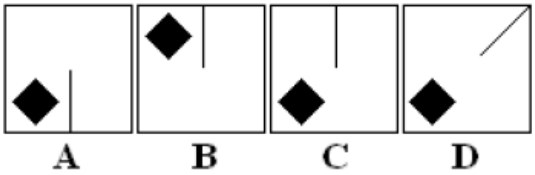
b. Looking for errors in real time.
c. Using a different way of thinking to approach situations, in which unnecessary details are hidden.
d. Using previous solutions to solve new problem situations.
e. Dividing a complex problem in little problems to solve the situation efficiently.

6. Which of the following situations represents the use of generalization?

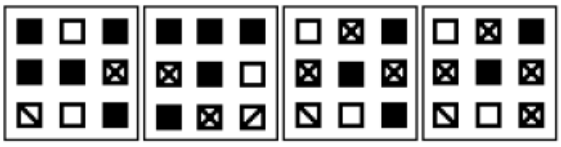
a. Making a step by step set of instructions to solve a situation
b. Looking for errors in real time.
c. Using a different way of thinking to approach situations, in which unnecessary details are hidden.
d. Using previous solutions to solve new problem situations.
e. Dividing a complex problem in little problems to solve the situation efficiently.

7. In the following sequence

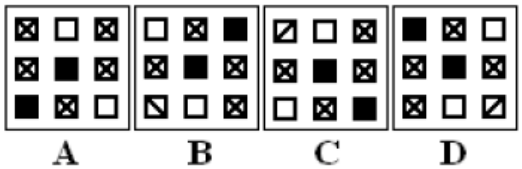
Which figure completes the series?




8. In the following sequence

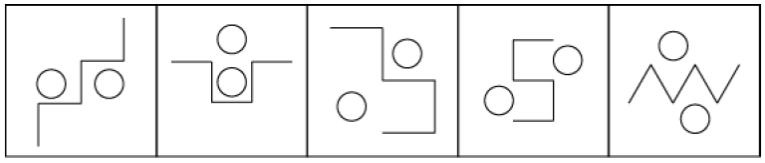


Which figure completes the series?

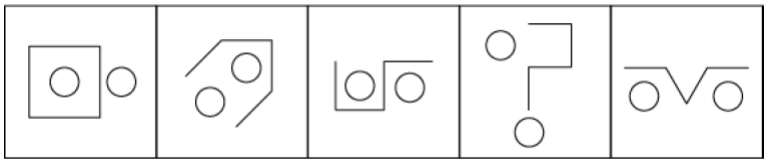


9. Which set does the Figure belongs to? Set

Set A



Set B



Figure

10. On a distant planet, the dominant carnivore, the zab, is nearing extinction. The number of zabs born in any year is one more than the (positive) difference between the number born in the previous year and the number born in the year before that.

    Example If 7 zabs were born last year and 5 the year before, 3 would be born this year

    Example 2: If 7 zabs were born last year and 10 the year before, 4 would be born this year.

    5 zabs were born in the year 2010 and 12 zabs were born in 2011. Predict the number of zabs born in 2009 and 2013.

    **Answer**: 2009: 2013:

11. A word search uses the following special symbols:

    ? Represents a single letter
    * Represents any number of letters, including no letters.

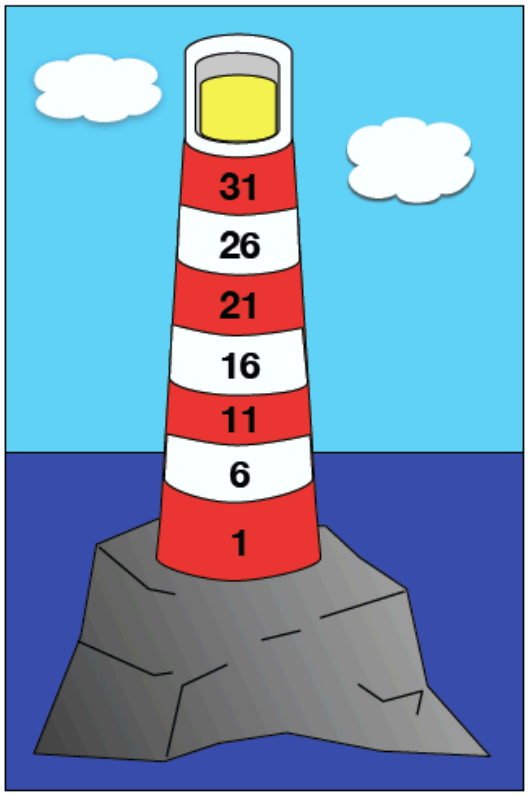
In order for a search term to match a word, it must represent the entire word from start to finish. For example, b * t matches bat but not bath.

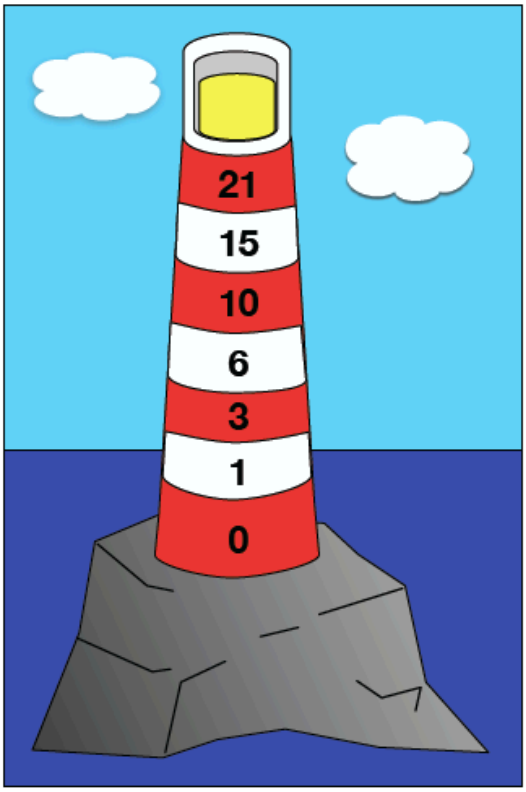
How many of the following words does c??s*ing match?

capsuling, causing, cashing, chasing, censoring, classifying, choosing, clustering, closings.

(A) 2 (B) 4 (C) 5 (D) 6 (E) 7

12. Write the umber in the light of the lighthouse that continues the pattern. L1: L2:




13. The following is a correct fragment of code (in Java) for sorting an array.

```
Public static void sort (int []a, int n)
 {
   for (int p=1; p =1; p <= n-1; p++)
    {
     for (int i=0; i < n-p; i++)
      {
       if (a[i] > a[i+1])
        {
         swap (a, i, i+1);
        }
      }
    }
 }
```

The following version has mistakes. Highlight the differences you can spot.

```
Public static void sort (int []a; int n)
{
  for (int p=1; p =1; p <- n-1; p++)
   {
    for (int i=0; i < n-p; i++)
     {
      if (a[i] < a[i+1])
       {
        swap (a, i, i+1).
       }
     }
   ]
}
```