Work in Progress: Incorporating the Engineering Design Process to Solve Real-Life Programming Problems in an Introductory Engineering Course

Dr. Aidsa I. Santiago-Román, University of Puerto Rico, Mayaguez

Dr. Aidsa I. Santiago-Román is an Associate Professor in the Engineering Sciences and Materials (CIIM) Department at the University of Puerto Rico, Mayagüez Campus (UPRM). Dr. Santiago earned a BA and MS in Industrial Engineering from UPRM and Ph.D in Engineering Education from Purdue University. Dr. Santiago has over 20 years of experience in academia and is currently the Department Head of the CIIM Department. She's also the founder and advisor of the first student chapter of the ASEE in Puerto Rico. Her primary research interests include investigating students' understanding of difficult concepts in engineering sciences, especially for underrepresented populations. She also works in the development and evaluation of various engineering curriculum and courses at UPRM applying the outcome-based educational framework.

Dr. Jairo Andres Agudelo, University of Puerto Rico, Mayaguez

I have a Bachelor's degree in Civil Engineering from the University of Quindío (Colombia, 2004), as well as a Master's and PhD from the University of Puerto Rico at Mayagüez (UPRM, 2014). My specialty lies in Structural Engineering. My teaching experience has been acquired throughout my academic and professional career. During the last two years, I have been teaching Algorithms and Programming, Material Mechanics I, and Engineering Graphics courses as an Assistant professor at the Department of Engineering Sciences and Materials at the UPRM. I was an instructor for five years and taught Mathematic Methods for Engineers in the UPRM Civil Engineering and Surveying Department. I was also an assistant professor at Caribbean University (CU) at Ponce, PR, where I taught Statics, Dynamics, Fluid Mechanics, and Structural Analysis courses. In the University of Quindío, I taught Statics and Physics I courses.

My academic life has given me the opportunity to develop different research projects with various themes such as seismic vulnerability and risk; structural behavior of building subjected to seismic, wind and water loads and seismic isolation devices for buildings. I am currently working in some research projects related to engineering education with professors from the department in order to develop this area in the university.

Arturo Ponce P.E., University of Puerto Rico, Mayaguez

Arturo Ponce has a BS in Computer Engineering and a MS in Electrical Engineering from UPR Mayaguez. He has a PhD in Computer Information Systems from Nova Southeastern University. He is an associate professor at the at the UPR Mayaguez Engineering Sciences and Materials Department. He has done has done institutional research work at UPR Mayaguez and also has worked in the ABET accreditation process for the School of Engineering .

WIP: Incorporating the engineering design process to solve real-life programming problems in an introductory engineering course

Abstract

For many students, the concepts involved in courses about algorithm and programming are very difficult to understand. Many professors pay more attention to the programming skills and rules that are not as critical for students in their academic career in engineering. As a result, students have high proficiencies for coding but are presenting difficulty in the process of understanding, analyzing, and solving problems, therefore being unable to transfer the acquired knowledge into real-life problems.

In this paper we present preliminary results of incorporating the engineering design process into the introductory engineering programming course to solve real-life problems. This methodology has been implemented for the past two years, so that students were able to develop problem-solving skills through the application of the design process and algorithmic concepts, resulting in the completion of a project that impacted various offices at the university and/or community industries. To complete this project, first students needed to identify a possible topic applicable for the course. Then they needed to understand the problem at the site and design a solution to satisfy their needs. Also, they generated a user manual that allows office personnel to make reference as needed, since once students graduate, they are not available to answer questions. Finally, students installed the software in a computer designated to be used and train the personnel to use the program.

Professors that implemented this approach perceived an increase in students' motivation throughout the course. To validate this assumption, a statistical analysis was conducted to compare students' performance on the course by comparing two groups: a control group that represents traditional teaching approaches and the experimental group that incorporated the design process in the methodology students' followed to design their solutions. Statistical results of overall students' evaluations for both groups validated these perceptions.

Introduction

Programming students and professors agree that topics covered in a computer programming introductory course are difficult for students to understand. Students find that programming concepts are too abstract and syntactic rules only make problems more difficult[1, 2, 3, 4]. Students' inability to understand the abstract nature of these programming problems and their lack of providing appropriate solutions make them feel afraid and unmotivated. The most important factor to consider when seeking solutions for these programming tasks is the teaching of algorithmic thinking and problem solving[5].

At the University of Puerto Rico, Mayagüez Campus (UPRM), the Algorithm and Programming course (A&P) offered at the Department of Engineering Sciences and Materials is comprised mostly of second year engineering students that have passed the pre-calculus course. The A&P course emphasizes the development of algorithms and its implementation at a high-level language. The department is forced to offer three different versions of the course, each one using

a particular language (VBA, Matlab, and C), as solicited by the degree-granting engineering departments.

Based on the author's experience teaching the course, we have identified certain deficiencies. First, many students face difficulties when learning the course material because of various situations such as: lack of awareness, general disinterest for their studies, struggle interpreting homework or assignments, difficulties when expressing themselves both written and verbally, lack of reading habits, little or no discipline for studying, little retention of acquired knowledge, and low grade reflection, independence, and/or generalization. These situations, when added to the fact that the course requires the use of mental processes that are generally complex and require creativity, ingenuity, and discipline, can cause a high desertion rate and a low retention rate. This is, consequently, reflected in the low passing rate, which is currently about 35%.

Second, the teaching method being implemented by many faculty has lost sight of the fact that students should learn the basic abstraction abilities and build up from there to more advanced tasks. According to Bloom's Taxonomy[6] there are six different levels of knowledge. The ability to program can reach up to the fifth level, synthesis, and yet students are asked to program without passing through the previous four levels. Furthermore, students find the programming concepts to be abstract and the rules of syntax make the problem even more difficult. As such, students feel afraid and with lack of motivation because of their inability to face the abstract nature of the problems and to obtain good programming results.

Another deficiency that we have identified is the teaching method. This course has been taught for many years by lecturing. This method focuses in the professor actively exposing the concepts and students passively taking notes. Therefore, this method does not allow for active student participation and does not develop teamwork skills that are needed in a professional setting. By having the professor be the main character in the classroom and students act as empty vessels waiting to be filled with information, students often lose interest in the matter altogether and oftentimes withdraw from the course or fail.

Due to the previously exposed deficiencies, this study aims to improve students' learning experience with the objective to develop basic abilities any professional engineer must have. These abilities include: (1) ability to understand the problem (take, mold, analyze), (2) propose effective solutions (reflect on an abstraction, define strategies, follow a process, apply a methodology, decompose in sub-problems), (3) manage languages in order to express a solution (do, understand, and respect syntax), (4) use tools to understand the languages (program, compile, execute, debug), (5) test valid solutions (understanding the concept of correction and testing), and (6) justify decisions (measuring, increasing), among others. We propose to incorporate the design process into the course as a tool for students to solve real-life problems related to programming concepts. Therefore, the research question guiding this study is: **What is the effect of incorporating the engineering design process in students' performance in an introductory programming course**?

In this paper we present preliminary results of this study where students completed a project that impacted various offices at the university and/or community industries. First, students needed to identify a possible topic applicable for the course. Then, they needed to understand the problem

at the site and design a solution to satisfy their needs. Finally, they had to install the program and train the selected personnel to use it.

Literature Review

Problem Solving

A solution to a problem involves finding a way to solve a difficulty, overcome an obstacle, and attend a clearly unknown purpose[7]. This is a cognitive process, which implies conscious and subconscious thinking. According to findings from our recent accreditation efforts, recent graduates find it difficult to solve real-life problems since it is very difficult for them to put the theory into practice. Educators and psychologists, with the objective of finding the best strategies to teach multiple levels, have developed many research studies about problem-solving skills. Therefore, professors must be able to teach successful problem solving methods, motivate students, and allow them to be more engaged in finding accurate solutions.[8] It has been found that traditional teaching focuses on increasing students' ability to memorize. Various researchers agree that teaching problem-solving is more efficient than the traditional method because it can result in better retention skills in the long run, and promotes the development of critical thinking abilities.[9, 10, 11, 12]

Furthermore, teaching algorithms should create the ability to propose, analyze, and solve problems through appropriate algorithm selections for particular and general purposes.[13] Also, algorithm courses should be organized around problem-solving techniques. Unfortunately, most modern courses and books focus on teaching programming languages and place problem-solving issues as those of secondary importance[14]. Modern teaching focuses on the instructor introducing a technique, showing how to solve a problem with this technique, and then asking students to repeat the procedure in a homework assignment or test problem. This focus allows students to apply and implement processes, but does not teach them to think, structure, and select alternative solutions, an aspect that is of great importance in real-life.

Since developing an algorithm is a multi-step process, we have established the following procedure for solving programming problems following the design process. It begins with a problem that needs solving. Then a solution is designed. The aforementioned solution is tested to see if it contemplates all possible cases and verifies the inappropriate cases in its solutions. Finally, if the solution meets the conditions, it is encoded with some kind of programming language. In most cases, the encoding is the last part of the process.

Design Process

The engineering design process consists of a series of steps to be followed with the objective of providing a solution to a problem that will result in the development of a new product or system. Pahl and Beitz[15] established the following steps for the design process: (1) Understand the problem, (2) Explore the problem, (3) Specify requirements, (4) Create alternative solutions, (5) Choose the best solution, (6) Build a prototype, and (8) Test and evaluate (8) Improve design.[15] The following section provides a detailed description of how this process was incorporated into the course and, specifically, into the project.

Methods and Results

Research Setting and Participants

This study is being conducted at UPRM, a Hispanic Serving and Land Grant institution. At the present time, the student population is comprised of 13,316 students, of which 12,283 of them are undergraduates (refer to Table 1). UPRM consists of the Division of Continuing Education and Professional Studies, the College of Agricultural Sciences, the College of Arts and Sciences, the College of Business Administration, and the College of Engineering.

The College of Engineering (CoE) offers a five-year degree program in Chemical, Civil, Computer, Electrical, Industrial, Mechanical, and Software Engineering. The College of Engineering represents a 36.9% of the student population and a 37.1% of the undergraduate population (refer to Table 1). The female graduate and undergraduate population at the CoE is 28.1% and 26.7% respectively. Similarly, at the CoE first and second year students represent 35.4% of the undergraduate student population. In 2016, the CoE awarded 534 degrees to undergraduate students, 31.5% of them to female students[16]. A summary of these statistics is presented in the table below.

Table 1. Student Population[17]

Classification	All	Female	Male	%Female	%Male
Undergraduate Level	12,283	5,658	6,625	46.1	53.9
Graduate Level	1,033	473	560	45.8	54.2
Graduate Engineering	359	101	258	28.1	71.9
Undergraduate Engineering	4,561	1,219	3,342	26.7	73.3
First Year Engineering	828	233	595	28.1	71.9
Second Year Engineering	786	214	572	27.2	72.8

Course Description

The Algorithms and Programming course (A&P) is one that belongs to the Department of Engineering Sciences and Materials (ESM) at UPRM. Students who take this course are mostly second year students that have approved the pre-calculus course. This course emphasizes the development of algorithms and its implementation at a high-level language. Since the department offers the basic engineering courses to all departments, it is forced to offer three different courses, each one using a particular language (VBA, MatLab, and C), as solicited by the engineering degree-granting departments. The percentage of students that have failed to obtain a passing grade (above 70%) has been around 30% during the last five years. This data compares negatively to Engineering Graphics Design (EGD), the previous engineering course required in their program of studies that has had a failure rate around 15% (refer to Figure 1).

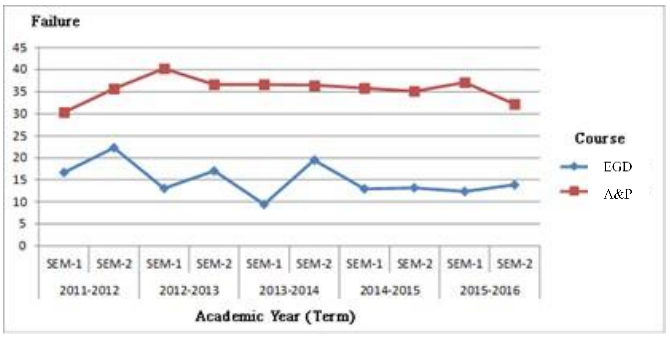

Figure 1. Failure percentage for A&P versus EGD[17]

Research Design

Proposed Teaching Methodology

Algorithmic thinking is a collection of skills that are allied to conceptualize and comprehend algorithms. According to Gerald Futsckek[18] it includes "the ability to (a) analyze given problems, (b) specify a problem precisely, (c) find the basic actions that are adequate to the given problem, (d) construct a correct algorithm to a given problem using the basic actions, (e) think about all possible special and normal cases of a problem, and (f) improve the efficiency of an algorithm".

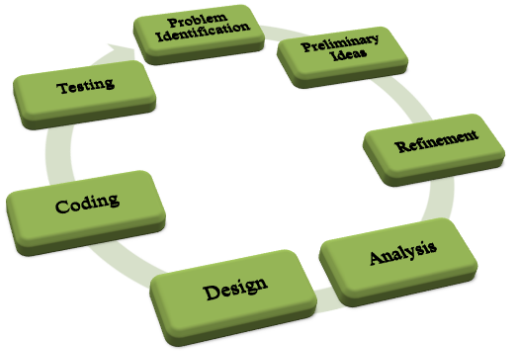

Figure 2 - Proposed Design Process for class project

In the A&P course, students should be able to develop problem-solving skills. To achieve this objective we incorporated the engineering design process into a project that requires the solution to a real-life problem as a course requirement. To complete this project, we established a modified design process that incorporates steps from the design process, as established by Pahl and Beitz[15], and those used to solve programming problems as explained in the previous sections. The steps of the proposed design process are as follows: (1) Problem Identification, (2) Preliminary Ideas, (3) Refinement, (4) Analysis, (5) Design, (6) Coding, and (7) Testing. A description of each step is provided in the following paragraphs.

a) *Problem Identification*. Most engineering problems are not clearly defined at the beginning and require identification before an attempt is made to solve them. In this step, students form groups of 3 to 5 members, where each student must identify a problem in their community or university, which they can solve through the development of a computer program in Excel using the VBA editor. Students begin with a blank sheet of paper, a pencil, and a few vague ideas. During most of this process, students communicate with community members and the professor to identify the statement of the problem and list its general requirements and limitations. Almost immediately, they should realize that they need additional information and data even before the project begins. Writing statements and making notes about the problem helps them "warm up" to the problem and begin the creative process. Subsequently, the professor and students choose one of the proposals from the group taking into consideration the requirements and limitations of each one.

b) *Preliminary Ideas*. The second step of the proposed design process is the development of ideas for possible problem solution. In this step the interface of the user's program is designed. A brainstorming session is required to collect ideas. This is the most creative step of the design process, and they are few restrictions and limitations at this stage. In this step, students should conduct a research of necessary information to solve the problem and define user necessities by answering the following questions: What does the user wish the program would do? How do they want the data to be entered? How and where do they want the data to be saved? How and where do they want to see the data? The answers to these questions will allow students to brainstorm ideas and design an interface that is more convenient for the user.

c) *Refinement*. In this step, several of the better user interface ideas are selected for refinement to determine their merits. Then, they are presented to the user so they can analyze them. The users decide which of the presented alternatives is the most appropriate to fulfill their needs.

d) *Analysis*. Once students have completed the previous steps, they must determine the process and restrictions in the design to obtain the desired output. Thus, students define how the computer manipulates the information to obtain the desired results. Therefore, the following elements must be defined: input and output variables, constants, conditions or limitations, type of data for each variable, input and output messages, and the procedure to be followed.

e) *Design*. The design requires the development of an algorithm to solve a problem. Design is the most important step in writing a program. In this step, students are able to understand uncertainties of the original problem that they would probably not realize until they had already completed major portions of the program. A finalized design allows relating the specifications with the final result to determine if the program executes as expected before continuing to the next phase. A good design divides a large problem into multiple simple problems (or modules) that are interconnected and can be reused in other programs, making the program more efficient and easier to understand. After completing the design, the correct operation of the algorithm must be verified through the appropriate manual tests. This allows students to know if there are logical errors in the proposed design.

f) *Coding*. In this step, students translate the algorithm into a computer program. They must convert each step from the proposed algorithm design into one or more VBA statements. Once completed, the result will be a program ready to be tested.

g) *Testing*. In this step, students will run the program, completing all its instructions and examining the logic by entering sample data to verify the output.

The proposed procedure is not a lineal process but rather an iterative one, which can be perfected by students, professors, and programmers according to the adjustments made based on the program needs so the final product can meet the initial requirements. This procedure is presented to students throughout the semester in various scenarios such as in-class problems, assignments, and exam problems.

Some of the project topics include registry of documents received and evaluated in an office, registry of people using the office facilities or resources, accounting of income and expenses in a company, accounting of personnel attendance and paychecks, registry of students' academic progress, etc. These topics were selected in a way that they could benefit not only students' learning processes but also the offices and/or organizations they impacted. The professors tried to maintain an equal level of difficulty among group projects. Some of the offices/organizations impacted were internal (from the university) and also external (usually from a family member).

During the semester, students had to develop a journal to keep record of every project activity they were involved. As engineers, documentation is crucial and it's an ability that needs to be developed. Also, each group meets with the professor every two weeks to keep track of their progress. The project evaluation is divided in stages, in accordance with the proposed design process, to facilitate their progress. Once the program is designed and tested, each group submits a written report that includes all the documentation for the program, but also a user manual to be handed to the "clients". Also they prepare an oral presentation to other students, faculty and the personnel from the offices they impacted. Once corrections are made, according to suggestions that occur during the presentations, they install the program in a designated computer as requested by their "clients", hand-in the user manual and train the corresponding personnel on the use of the program.

Data Collection and Analysis

This study compared the passing rate for two populations. One population consisted of students that took a regular programming course and the other population consisted of students that took a programming course that included real-life problem solving as a way of delivering the material. Visual Basic for Applications (VBA) was the programming language used to teach both populations.

The data was collected from 2012-2013 spring semester to 2016-2017 fall semester, a total of eight semesters. Table 2 and Table 3 show the passing rates for each group. The passing rate was calculated dividing the number of students that obtained a passing grade in the course by the total number of students registered.

Table 2. Population that took VBA with traditional learning (lecturing)

Academic Year	Semester	A, B or C	Registered	Passing rate (%)
2012-2013	Spring	18	50	36.00
2013-2014	Fall	11	28	39.29
2013-2014	Spring	7	36	19.44
2014-2015	Fall	19	51	37.25
2014-2015	Spring	13	48	27.08
2015-2016	Fall	19	53	35.85
2015-2016	Spring	32	66	48.48
2016-2017	Fall	23	62	37.10

Table 3. Population that took VBA with real-life problem solving

Academic Year	Semester	A, B or C	Registered	Passing rate (%)
2012-2013	Spring	33	38	89.47
2013-2014	Fall	69	81	85.19
2013-2014	Spring	42	52	80.77
2014-2015	Fall	85	106	80.19
2014-2015	Spring	45	58	77.59
2015-2016	Fall	77	104	74.04
2015-2016	Spring	61	75	81.33
2016-2017	Fall	81	99	81.82

An independent two-sample t-test with unequal variances was conducted to verify that the results were not obtained by chance. The following statistical hypothesis were formulated:

- H_0 Passing rate for both populations is equal
- H_1 Passing rate for both populations is unequal

For a two-tailed t-test, a p-value < $\alpha/2 = 0.025$ (assuming $\alpha = 0.05$) meaning that there is a statistical difference and the null hypothesis can be rejected[19]. A P-value equal to 3.70E-08 was obtained in this test (Table 4). This means that the alternative hypothesis can be adopted.

Table 4. Two-Sample t-test assuming unequal variances

Results	Population that took VBA with real-life problem solving	Population that took VBA with traditional learning
Mean	81.30	35.06
Variance	21.46	73.72
Observations	8	8
Hypothesized Mean Difference	0	
Df	11	
t Stat	13.40	
P(T<=t) two-tail	3.70E-08	
t Critical two-tail	2.20	

Conclusions and Implications

The goal of this study was to provide students with an academic experience that will assist them in developing problem solving skills and professional habits and norms throughout the incorporation of the design process for the solution of real-life problems that consequently is of benefit to the community. Our preliminary results look very promising and provide evidence that this type of activity improved student performance. Furthermore, the authors have noticed an

improvement in students learning experiences, which reflects in their behavior. In other words, students are more interested and engaged in the course because not only were they able to meet the course requirements, but at the same time they demonstrated their ability to think like an engineer. This is evidenced in the increased overall course performance, as compared with the control group, even though the project represents only 20% of their final grade.

Integrating real-life problems into a project for the algorithmic and programming course has proven to be an effective method to improve the learning experience. Specifically, faculty perceived that students were able to develop strong and effective values of project management, teamwork, leadership, and civic participation. The projects completed by students allowed them to apply the knowledge acquired in the course, interact with the community, observe their needs and design an effective solution to meet those needs.

Informal feedback from students indicated that the project made a positive impact on them and increased their motivation in the course. This and the increased passing rate of the course gives us reason to believe that the changes made on the original course had a positive effect. However, at this time, it is too early to determine with certainty the causality of this result. In the future we will design surveys that allow quantitative measure of satisfaction of the project sponsor. We also plan to conduct a study to document students' perceptions on the impact of the project experience in their academic and professional careers. We also intend to design pre-test and post-test that permits us to measure the level of knowledge reached by each group and to develop surveys as an instrument of course assessment. These tools can demonstrate possible differences in the impact of this course on students with different profiles.

Acknowledgements

We would like to acknowledge the Department of Engineering Sciences and Materials and the College of Engineering at the University of Puerto Rico at Mayagüez, for providing the investigators with the time and the financial support of teaching assistants to conduct this study. Also special thanks to the MS students Mr. Heriberto L. Pujols, Mr. Christian G. Hernández, and Ms. Nathalia Ospina who served as TAs and assisted in the data collection and assessment of the educational activities assigned to students enrolled in the A&P course during the past years.

Bibliography

1. Gomes, A., & Mendes, A. J. (2007). An environment to improve programming education. *Proceedings of the 2007 International Conference on Computer Systems and Technologies*, 88.
2. Smith, P. A., & Webb, G. I. (2000). The efficacy of a low-level program visualization tool for teaching programming concepts to novice C programmers. *Journal of Educational Computing Research*, 22(2), 187-215.
3. Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137-172.
4. Jenkins, T. (2002). On the difficulty of learning to program. *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences* (4), 53-58.
5. Konecki, M., & Mrkela, V. (2014). Algorithmic thinking and animated interactive presentation of sorting algorithms in education of students. *Proceedings of the 17th International Multiconference Information Society-Education in Information Society*, 105-112.
6. Bloom, B.S. (Ed.). Engelhart, M.D., Furst, E.J., Hill, W.H., Krathwohl, D.R. (1956). Taxonomy of Educational Objectives, *Handbook I: The Cognitive Domain. New York: David McKay Co Inc*.
7. Polya, G., "How I solve it: A new aspect of mathematical method", New Jersey: Princeton University Press, 1973.
8. Giannakopoulos, A. (2012). Problem solving in academic performance: A study into critical thinking and mathematics content as contributors to successful application of knowledge and subsequent academic performance. *Unpublished PhD*, UJ, South Africa.
9. Lunyk-Child, O. I., Crooks, D., Ellis, P. J., Ofosu, C., & Rideout, E. (2001). Self-directed learning: Faculty and student perceptions. *Journal of Nursing Education*, 40(3), 116-123.
10. Stepien, W. J., Gallagher, S. A., & Workman, D. (1993). Problem-based learning for traditional and interdisciplinary classrooms. *Journal for the Education of the Gifted*, 16(4), 338-357.
11. Gallagher, S. A., Stepien, W. J., & Rosenthal, H. (1992). The effects of problem-based learning on problem solving. *Gifted Child Quarterly*, 36(4), 195-200.
12. Hmelo, C. E., & Ferrari, M. (1997). The problem-based learning tutorial: Cultivating higher order thinking skills. *Journal for the Education of the Gifted*, 20(4), 401-422.
13. Torrey, L. (2012, July). Teaching problem-solving in algorithms and AI. *Third Symposium on Educational Advances in Artificial Intelligence*.
14. Cormen, L.T, C. E., Rivest, R. L., & Stein, C. (2001). Introduction to algorithms (Vol. 6). Cambridge: MIT Press.
15. Pahl, G., & Beitz, W. (2013). *Engineering design: a systematic approach*. Springer Science & Business Media.
16. Tasas de graduación, retención y terminación a tiempo, 2016. http://www.uprm.edu/p/oiip/tasas_de_graduacion_y_retencion
17. Datos Históricos, 2016. http://www.uprm.edu/p/oiip/tasas_de_graduacion_y_retencion

18. Futschek, G. (2006, November). Algorithmic thinking: the key for understanding computer science. *International Conference on Informatics in Secondary Schools-Evolution and Perspectives*, 159-168. Springer Berlin Heidelberg.
19. Brase, C. H. & Brase, C. P. (2007). *Understanding basic statistics*. (4th ed.). Boston, MA, USA: Houghton Mifflin Company.