A. Rojas-López and F. J. García-Peñalvo, "Initial learning scenarios based on the computational thinking evaluation for the course Programming fundamentals at INACAP," in TEEM'19 Proceedings of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality (Leon, Spain, October 16th-18th, 2019), M. Á. Conde-González, F. J. Rodríguez-Sedano, C. Fernández-Llamas and F. J. García-Peñalvo, Eds. ICPS: ACM International Conference Proceedings Series, pp. 6-12, New York, NY, USA: ACM, 2019. doi: 10.1145/3362789.3362802.

Initial learning scenarios based on the computational thinking evaluation for the course Programming fundamentals at INACAP

Arturo Rojas-López
Information Technologies Division,
Technological University of Puebla
Puebla, México
arturo.rojas@utpuebla.edu.mx

Francisco J. García-Peñalvo
GRIAL Research Group,
Research Institute for Educational Sciences, University of Salamanca
Salamanca, Spain
fgarcia@usal.es

ABSTRACT

The paper's objective is present the design and the planning of initial learning scenarios for the course Programming Fundamentals, from the evaluation of computational thinking to new students of the careers Computer engineering and Programmer analyst of the Technological University of Chile and Training Center Technical respectively at INACAP, to favor the motivation and autonomy of study through the recognition of skills and the use of the instructional design of the face-to-face course. The proposal is based on correspondence with three of five change trends that integrated the educational model. Regarding the Knowledge society, promote recognition of the individuality of the student as a person who will do university studies, that is, the scenarios respond to the fact that each person learns differently. In the Training of competences, contribute with preventive actions that the teacher communicates when there is a lack of specific skills. Finally, in the Flexibility and articulation, provide a diagnostic tool that favors the recognition of previous competences to have an articulated beginning of studies based on the needs of the student. Consequently, contribute to the INACAP´s educational model.

KEYWORDS

Computational thinking, Problem solving, Higher education, Programming teaching, Programming Learning.

1 Introduction

INACAP´s educational model [1] is comprised by five trends of change in the functions and structures of higher education: 1) Knowledge society, 2) Training of competences, 3) Flexibility and articulation, 4) Coverage and 5) Quality management. In the Knowledge society, the focus of education is placed on the person, recognizing the heterogeneity of his profile. In the Training of competences, the demonstration instances of performances must allow the teacher to provide feedback in a timely manner to the students with the aim of ensuring their ability to act professionally once they have graduated; from the point of view of training, it implies having progressive learning architectures, diversification of pedagogical methodologies and the protagonist assignment to the student. In the Flexibility and articulation, previous learning must be recognized not only in formal educational processes, to articulate learning according to the needs and interests of the student. Echegaray et al. [2] describe several factors that students face in order to maintain a university academic life: "students discouragement, confusion regarding the choice of degree, information lack about university life or the plans and content of the degrees, confusion with the own design of the university structure, deficiencies in the previous academic formation, insecurity feelings about the own capacities, etc.". In similarity to other Higher Education Institutions, INACAP has a first-year retention percentage (2016-2017) for the Technical Training Center of 70.2%, and for the Technological University of 67.0% [3]. In particular, the learning-teaching of computer programming is not a simple activity for student-teachers and is more complicated if the skills insecurity or abilities in problem solving is not detected before the beginning of a course. Selby [4]

particularly addresses the learning problem, noting that some of the identified reasons are: "inadequate understanding of how a computer model works, an inability to master reading, tracking and code writing, as well as, understanding high level concepts such as the design."

The frustration that can cause students during the first months of study, not to be able to create basic *software* solutions to problems in the industry or labor market, can be the main factor in making the decision to school dropout. The demand of programmers is still required in the economic system of any country and its diversity of industries: automotive, embedded devices, academia, business, advertising and payment technologies [5], particularly the United States Bureau of Labor Statistics [6] forecasts a 24% increase from 2016 to 2026; so the relevance of acquired skills and their functionality in professional practice reflects the success of a university educational model. In addition, we must consider the process of job automation that will require the professional activity of data analysts, experts in artificial intelligence and machine learning [7].

The initial concept of computational thinking was introduced by Wing in 2006 [8]. Now days, computational thinking is a type of analytical thinking, a set of cognitive and metacognitive strategies paired with processes, skills and methods of computer science (analysis, abstraction, decomposition, heuristic reasoning, planning, programming, model, pattern recognition, algorithm). Its essence is to think about data and ideas, and to use and combine these resources to solve problems, design systems and understand human behavior, in such a way that a computer - human or machine - can effectively carry out the solution [9-20].

The definition that serves as support to the proposed research is Selby´s study [4] which includes the skills of abstraction, decomposition, algorithmic design, generalization and evaluation. The studio detonates a very important element because it explores the relationship between computational thinking, teach programming and learning levels through the Bloom´s Taxonomy. Initially it develops a definition of the five skills.

- Generalization: the ability to express the solution of a problem in generic terms, which can be applied to different problems that share some of the same characteristics as the original problem.
- Decomposition: fractioning into smaller pieces, easy to solve, parts of a problem.
- Abstraction: ability to decide which details of a problem are important and which details can be omitted.
- Algorithmic design: ability to create a set of instructions that indicate step by step the solution of a problem for a device.
- Evaluation: ability to recognize and determine the scope of carrying out processes, in terms of efficiency and use of resources.

Subsequently, Selby maps the cognitive domain of Bloom's Taxonomy to concentrate it in the following way: in the *Application* level the Generalization ability, in the *Analysis* level the Abstraction and Decomposition abilities, in the *Synthesis* level the Algorithmic design skill and in the *Evaluation* level corresponds the ability of the same name. Finally, she relates programming knowledge (data types, problem analysis, algorithmic design, program creation, functionality, execution and evaluation of results) with the mapping described above (see Table 1).

Table 1: Learning levels mapping, skills and programming knowledge

Learning level	Computational thinking skills	Programming knowledge
Application	Generalization	Functionality
Analysis	Abstraction Decomposition	Data type Analysis of a problem
Synthesis	Algorithmic design	Design and creation of algorithms-programs
Evaluation	Evaluation	Test, evaluate algorithms-programs

Taking as a reference the face-to-face subject Programming Fundamentals, first course in both curricular meshes for software development (Computer Engineering / Programmer Analyst), where the description indicates a "practical subject oriented to develop the logical thinking of the student through the analysis and problem solving" can be considered an evaluation of computational thinking to determine the skills that possess a student in problem solving[21], and obtain initial learning scenarios that represent a first academic teacher's guide; with the objective of favoring the motivation and autonomy of study, in addition to reducing the school dropout through preventive actions instead of corrective ones during the development of the subject.

The general objective is checking the determination of initial sceneries of learning that the student may face in the presential modality, considering the assessment of computational thinking skills of new students, which benefits the teacher of the course Programming fundamentals to reduce dropout. The paper has the next organization. First, we present the explication about experiment design, this is, how relationship the course content, knowledge and skills with computational thinking evaluation. Second, we introduce the learning scenarios based with evaluation results. Besides, we propose surveys with aim of measure the adequate design of learning scenarios during the face-to-face course. Finally, in conclusion section, we explain the benefits for INACAP if experimental design is used in next initial course at 2020 year.

2 Methodology

2.1 Experiment design

The face-to-face course Programming Fundamentals has five learning units and 90 hours assigned for the 18-week semester. Due to INACAP's educational model, the instructional design integrates online work hours (72 in-person and 18 online for the course), the distribution hours detail, weeks, weighting and evaluation criteria number is indicated in Table 2. In each unit, it is described the expected learning (see Table 3) and a list of the evaluation criteria (see Table 4 to Table 8), for each subject there are three evaluative actions: diagnostic, formative and summative; the first two are suggested activities that will help improve the learning outcomes and will reveal student progress at different times of the subject, summative evaluation is mandatory and is carried out as stipulated by the course. Each learning unit brings together a percentage of the summative evaluation, the sum of the five in the course generates 80%, the 20% missing is specified by discretion for the teacher assessing presentations, projects or extra work.

Initially, a knowledge relation of the subject Programming fundamentals with the computational thinking skills is established as indicated in Table 9 [22,4]. In each relation, the justification and the evaluation criteria are indicated, an impact is determined, in addition to the name of the selected reagent [23-25].

Table 2: Planning the Programming fundamentals course

Learning unit	Hours			Weeks	Evaluation criteria / Weighing
	Face-to-face	Online	Total		
1. Data processing Fundamentals	12	3	15	3	5/10%
2. Control structures in DFD	16	4	20	4	6/20%
3. Control structures in pseudocode	20	5	25	5	6/20%
4. Array structure	12	3	15	3	5/15%
5. Subroutines	12	3	15	3	5/15%

Table 3: Expected learning per unit learning

Learning unit	Expected learning
1	Solve data processing problems, applying storage principles and truth tables.
2	It represents graphically the solution of a problem through diagrams data flow applying control structures.
3	Develop algorithms in pseudocode, applying control structures in the solution of a posed problem.
4	Develop algorithms in pseudocode, using Array one-dimensional and two-dimensional in the solution of a problem raised
5	Develop basic algorithms in pseudocode, through subroutines in the solution of a problem raised

Table 4: Evaluation criteria and activities, learning unit 1

Evaluation criteria	Activity - modality
1.1.1. Identifying input data, process and output from a processing problem of data. 1.1.2. Applying the stages of the methodology of Polya in the analysis of the solution posed problem. 1.1.3. Considering entry operations, processes and data output in memory. 1.1.4. Applying logical operators in the solving processing problems data.	- Diagnostic evaluation – face-to-face - Training activity 1 - online - Exercise guide 1 / formative evaluation - face-to-face - Training activity 2 - online - Exercise guide 2 / formative evaluation - face-to-face - Summative evaluation - face-to-face

2.2 Learning scenarios

The interpretation of the results obtained with computational thinking assessment is determined by the following 8 scenarios that the student may face in the course modality. The number of scenarios corresponds to the cases that represent from possible talents detection (five correct reagents) to the lack of skills for problem solving (five incorrect reagents). They are considered scenarios where possible gaps are established from the beginning, middle or end of the course with respect to the learning units with the corresponding reagents. Therefore, the proposed scenarios are not all right responses combinations of five reagents, only those according to the relationship with the learning unit. Units 1 and 2 have a basic level and great importance for the course because formative content and concepts. Unit 3 represents a medium level of learning when reviewing the concept of control structures in pseudocode. Finally, units 3 and 4 are associated with a high level of learning within the course. Table 10 shows the determination of the scenarios based on was previously commented.

The first scenario corresponds to the students who exhibit skills in problem solving, so it is considered that they will not have problems with the learning contents of the course and may have a high degree of autonomy. The second scenario determines the opposite case of the first. From the third to the eighth, the correct

reagents are taken into account according to the learning unit which it was related.

Scenario 1. 5 correct answers

The student will not have any problems to accredit following the instructional design, doing the face-to-face activities and those established for the subject online through the platform, that is, use of the training activities, learning guide and exercises, as well as, the evaluations formative and summative of each learning unit. An alternative scenario for the student, if it was possible, could be to advance the development of the subject at his own pace and only receive the guidance of the teacher when he requests it. Finally, he can also be a good candidate to take the course online.

Scenario 2. Incorrect in all items

There is the possibility that the student does not have skills for the study of the subject and in the unit of learning three it is difficult to understand the control structures and algorithmic design when creating algorithms in pseudocode, so it is recommended in meeting face-to-face teacher-directive-student assess the vocational profile. It is very sure that a classroom course is the best option for the student.

Scenario 3. Incorrect Kangaroo and Jumping puddles exercises.

The student requires observation and classroom teacher work so from initial units acquire meaningful learning. The student could accredit the face-to-face course without inconvenience taking into account to reinforce at the time (in week 4 of the course), in person or through the platform, the second learning unit, Control structures in DFD, because it would be their first contact with the control structures (sequential, decision and repetition) for the definition of algorithms, coding extra exercises to the corresponding guide that solve a problem using such structures. Otherwise, it may be difficult from the learning unit 3 and its accreditation is difficult.

Scenario 4. Correct Kangaroo and Jumping puddles exercises

The student exhibits basic skills for the creative activity of algorithm design. Possibly he will not have any problems to accredit the evaluation criteria in the first three weeks following the instructional design, that is, doing the face-to-face and online activities established for the subject in the platform. From the fourth week, work a special attention with a practice and feedback of extra exercises in the learning units two and three in person, thus also guarantee the prevention of some problems in the last two units of learning.

Scenario 5. Beavers exercise correct

The student requires observation and classroom teacher work so that from initial units acquire meaningful learning. There will be reinforce evaluation of arithmetic, logical and relational expressions considering the hierarchy of operators by means of coding exercises using calculation operations and conditional on the learning unit two (week 4 of the course). The above to prevent problems when addressing the learning unit 4 and face without conflict the first increase in complexity in the course.

Scenario 6. Beavers exercise incorrect

If it was possible, student could advance the development of the subject at his own pace and only receive the guidance of the teacher when he requests it through the platform itself in the first two units of learning, otherwise, follow the indicated planning in the face-to-face course. The student may have problems with the learning unit 3, Control structures with pseudocode, so it would be convenient to reinforce the algorithmic design with extra exercises to the corresponding guide through the platform or in person when arriving at this learning unit (in week 8 of the course), thus guaranteeing the prevention of some problems in the last two learning units.

Scenario 7. Correct Spies and Mobile exercises

The student requires observation and classroom work of teacher so that from initial units acquire significant learning, in addition to care that material and activities are carried out punctually based on the time given in the instructional design may also require sessions counseling with the teacher, so they should be encouraged so that the student's confidence increases with respect to the ability to create programs, and planned so that they do not represent extra work of the teacher outside the classroom hours. Otherwise, it may be difficult from the learning unit 3 and its accreditation is difficult.

Scenario 8. Incorrect Spies and Mobile exercises

The student has basic, but necessary skills for learning, particularly computer programming, so the instructional design created is favorable, but with a regular face-to-face assessment in the laboratory practices designed by the teacher.

It is advisable conduct two surveys to obtain data to validate the hypothesis with the questions indicated in Table 11 and Table 12, the first to be answered in week 9 of the course and the second at the end of the semester by students of the experimental groups.

Table 5: Evaluation criteria and activities, learning unit 2

Evaluation criteria	Activity - modality
2.1.1. Applying decision structures in the solution of the problem. 2.1.2. Incorporating logical operators in the solution of the problem. 2.1.3. Using repetition structures in solving the problem. 2.1.4. Considering the validation of data in the solution of the problem. 2.1.5. Making the trace of the proposed solution.	- Diagnostic evaluation - face-to-face - Guide to exercises 3 / formative evaluation - face-to-face - Forum - online - Exercise guide 4 / formative evaluation - face-to-face - Summative evaluation - face-to-face

Table 6: Evaluation criteria and activities, learning unit 3

Evaluation criteria	Activity - modality
3.1.1. Applying decision structures in the solution of the problem. 3.1.2. Using logical operators in the construction of algorithms. 3.1.3. Incorporating repetition structures in algorithms in Pseudocode. 3.1.4. Considering the validation of data in the solution of the problem. 3.1.5. Making the trace of the proposed solution.	- Diagnostic evaluation - face-to-face - Training activity - online - Exercise guide 5 / formative evaluation - face-to-face - Forum - online - Summative evaluation - face-to-face

Table 7: Evaluation criteria and activities, learning unit 4

Evaluation criteria	Activity - modality
4.1.1. Entering data in array one-dimensional and two-dimensional. 4.1.2. Doing a tack about array one-dimensional or two-dimensional. 4.1.3. Doing searches in array one-dimensional or two-dimensional. 4.1.4. Doing the trace of the solution proposal.	- Diagnostic evaluation - face-to-face - Training activity - online - Exercise guide 6 / formative evaluation - face-to-face - Forum - online - Summative evaluation - face-to-face

Table 8: Evaluation criteria and activities, learning unit 5

Evaluation criteria	Activity - modality
5.1.1. Incorporating exchange of parameters to the subroutine 5.1.2. Incorporating the return of data to main program. 5.1.3. Doing calls to subroutines created. 5.1.4. Doing the trace of the solution proposal.	- Diagnostic evaluation - face-to-face - Exercise guide 7 / formative evaluation - face-to-face - Forum - online - Summative evaluation - face-to-face

Table 9: Relationship of computational thinking skills with learning units

Learning unit	Skill Computational Thinking / reactive - Justification
1. Data processing fundamentals	Abstraction / Kangaroo - Abstraction helps to determine the data that contribute to the resolution of some problem as indicated by the expected learning of the unit. It will impact on the evaluation criteria: 1.1.1, 1.1.3, 1.1.4
2. Control structures in DFD	Evaluation / Jumping puddles - The evaluation allows recognize and determine the scope of execution of the data flow diagrams in the resolution of a problem that is part of the expected learning of the unit. It will impact on the evaluation criteria: 2.1.5
3. Control structures in pseudocode	Algorithmic design / Beavers - The algorithmic design allows the development of algorithms in pseudocode (or programming language) that is the objective of the expected learning of the unit. It will impact on the evaluation criteria: 3.1.1, 3.1.2, 3.1.3, 3.1.4
4. Array structure	Generalization / Spies - The management of arrays represents the first increase in the complexity of the algorithmic design, because the student has visualized the usability of the structure to solve problems where previously he had created a solution, but now he has incorporated the understanding of similar characteristics, so he can generalize a solution optimally. It will impact on the evaluation criteria: 4.1.2, 4.1.3
5. Subroutines	Decomposition / Mobile - The creation of subroutines or functions represents the ability to fragment (decompose) a problem into functional blocks and smaller size. It will impact on the evaluation criteria: 5.1.1, 5.1.2, 5.1.3

Table 10: Determination of learning scenarios

Level	Ability Unity Reagent	Scenarios 1 - correct 0 - incorrect							
		1	2	3	4	5	6	7	8
Basic	Abstraction 1 Kangaroo	1	0	0	1	0	1	0	1
	Evaluation 2 Jumping of puddles	1	0	0	1	0	1	0	1
Medium	Algorithmic design 3 Beavers	1	0	1	0	1	0	0	1
High	Generalization 4 Spies	1	0	1	0	0	1	1	0
	Decomposition 5 Mobile	1	0	1	0	0	1	1	0

Table 11: Half semester survey

Question	Response option
The work modality seems appropriate with your expectation of learning	Yes Not
Do you know the objectives that have to reach or are you clear about the knowledge and what you should know to do at the end of the course?	Yes Not
Do you feel lost using the platform, do not know what to do and for what?	Yes Not
Of the activities contained in the platform Which one gave you more?	Multiple choice - Diagnostic evaluation - Training activity - Exercise guide / formative evaluation - Forum - Summative evaluation
What action do you suggest to improve learning or do you agree with your learning environment?	Open

Table 12: Survey at the end of semester

Question	Response option
Select the learning criteria which you are familiar - Multiple selection Identifying input data, process and output from a processing problem of data. Applying logical operators in the solving processing problems data. Applying decision structures in the solution of the problem. Using repetition structures in solving the problem. Considering the data validation in the problem solution. Making the trace of the proposed solution. Using logical operators in the construction of algorithms. Incorporating repetition structures in algorithms in Pseudocode. Entering data in array one-dimensional and two-dimensional. Doing a tack about array one-dimensional or two-dimensional. Doing searches in array one-dimensional or two-dimensional. Incorporating exchange of parameters to the subroutine Incorporating the return of data to main program. Doing calls to subroutines created.	
Was the learning modality adequate to acquire the competences of the course?	Yes Not
Was the evaluation of your skills at the beginning of the semester an appropriate activity to determine the best initial learning environment?	Yes Not
What recommendation do you have for future generations about how to learn the content of the course?	Open

3 Conclusions

The learning scenarios for the new students of the course Programming fundamentals are proposed to benefit the INACAP´s retention percentage. Its design is based on the study of the learning units' contents and the material available on the platform, that is, considering the evaluation criteria and the instructional design of the course.

The ideal scenario for the intervention of the experiment in classrooms, is one where all students who begin the course answer computational thinking evaluation online as an activity during the first week. Subsequently, determine the initial scenarios that teachers-students may face during the course from the correct reagents obtained by each of the young people. Stable control and experimental groups. Experimental groups have to perform preventive actions or conditions that consider the recommendations of each scenario for its section of the teacher. During the course, register and measure the correspondence of the scenario with the results of the evaluation criteria of the students in their respective moment, that is, verify that in the control groups a percentage of school dropout and accreditation must be observed in correspondence with the historical data of the Computer and Telecommunications area of the last 3 to 5 years; for the experimental groups, it is expected to favor the motivation and study autonomy to accredit the course and consequently decrease the percentage of school dropout.

ACKNOWLEDGMENTS

The present work is carried out within the Doctorate program in Education in the Knowledge Society of the University of Salamanca, Spain [26-28], the GRIAL research group [29,30], as well as, the support of the Program Pacific Alliance at Ministerio de Relaciones Exteriores – Chile, INACAP and Technological University of Puebla.

REFERENCES

[1] Model Educational Institutional 2015 - INACAP, available on line: http://www.inacap.cl/web/acerca-de/Modelo-Educativo-2015.pdf, last consultation in May 2019.

[2] G. Echegaray, N. Barroso, I. Laskurain, K. Zuza, J.I. Barragués, (2017) Qualitative research for the improvement of the academic results in the first year in the Engineering degrees of the School of Engineering of Gipuzkoa. IV International Congress on Learning, Innovation and Competitiveness-CINAIC (Zaragoza 4-6 October 2017). DOI: http://dx.doi.org/10.26754/CINAIC.2017.000001_093.

[3] SIES- Service of Information of Education Superior (2019) Methodology Data published in seeker of institutions, www.mifuturo.cl 2018-2019, available on the Internet: http://portales.inacap.cl/transparencia/.

[4] C.C. Selby, (2015) Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. In Proceedings of the Workshop in Primary and Secondary Computing Education (WiPSCE '15). ACM, New York, NY, USA, 80-87, DOI: http://dx.doi.org/10.1145/2818314.2818315.

[5] K. Mazaika (2017) Will the Demand for Developers Continue to Increase? Quora contributor on the Internet at https://www.forbes.com/sites/quora/2017/01/20/will-the-demand-for-developers-continue-to-increase/#f90ee0133ee9 (visited May 10, 2019).

[6] Bureau of Labor Statistics, US Department of Labor, Occupational Outlook Handbook, Software Developers, on the Internet at https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm (visited May 10, 2019).

[7] M.A. García (2019) Automation: this is the battle between work and technology. Retina Magazine - May 25, 2019, available on the Internet: https://retina.elpais.com/retina/2019/05/24/tendencias/1558680372_855666.html?id_externo_rsoc=TW_CM (accessed May 28, 2019).

[8] J.M. Wing (2006) Computational Thinking. Communications of the ACM 49, 3, 33-35. DOI: http://dx.doi.org/10.1145/1118178.1118215.

[9] M. Romero, A. Lepage and B. Lille (2017) Computational Thinking development through creative programming in higher education. International Journal of Educational Technology in Higher Education 2:42. DOI: http://dx.doi.org/10.1186/s41239-017-0080-z.

[10] Q. Gao (2014) The computational thinking-oriented inquiry teaching mode for advanced programming language course. Bio Technology: An Indian Journal. Volume 10, Issue 12. pp. 6587-6595. ISSN: 0974-7435.

[11] Z. Yinnan and L. Chaosheng (2012) Training for Computational Thinking Capability on Programming Language Teaching, 7th International Conference on Computer Science and Education (ICCSE), Melbourne, VIC, 2012, p. 1804-1809. DOI: http://dx.doi.org/10.1109/ICCSE.2012.6295420.

[12] C. Zhang, X. Chen, and L. Li (2011) Research of VB programming teaching based on the core of computational thinking ability training, 6th International Conference on Computer Science & Education (ICCSE), Singapore, 2011, pp. 1260-1263. DOI: http://dx.doi.org/10.1109/ICCSE.2011.6028861.

[13] W. Huang, Z. Deng, and D. Rongsheng (2009) Programming Courses Teaching Method for Ability Enhancement of Computational Thinking, International Association of Computer Sciences and Information Technology - Spring Conference, Singapore, 2009, p. 182-185. DOI: http://dx.doi.org/10.1109/IACSIT-SC.2009.52.

[14] G. Chen (2017) Programming Language Teaching Model Based on Computational Thinking and Problem-based Learning. 2nd International Seminar on Education Innovation and Economic Management (SEIEM 2017). Atlantis Press. DOI: http://dx.doi.org/10.2991/seiem-17.2018.31.

[15] Z. Ni (2017) Discussion on Case Teaching Method Based on Computational Thinking in Programming Teaching. International Conference on Social Science, Education and Humanities Research (ICSEHR 2017). DOI: http://dx.doi.org/10.2991/icsehr-17.2017.9.

[16] L. Ying and L. Pingping (2017) Research on the teaching of programming language based on Computational Thinking. Proceedings of the 2017 International Conference on Social Science, Education and Humanities Research. DOI: http://dx.doi.org/10.2991/icsehr-17.2017.17.

[17] C. Zhi- Mei and L. Xiang (2016) The PBL teaching method based on computational thinking in C programming. 2 nd International Conference on Modern Education and Social Science. ISBN: 978-1-60595-346-5, pp.405-409.

[18] A. Xia (2016) On the Basis of the Program Design Teaching and Research of Cultivation of Computational Thinking Ability. Proceedings of the 2nd International Conference on Social Science and Higher Education. Atlantis Press. DOI: http://dx.doi.org/10.2991/icsshe-16.2016.83.

[19] P. Compañ-Rosique, R. Satorre-String, F. Llorens-Long, and R. Molina-Carmona (2015) Teaching to schedule: one way direct to develop the thinking computer. Journal of Education to Distance, 46 (11). DOI: http://dx.doi.org/10.6018/red/46/11.

[20] G. Michaelson (2015) Teaching programming with computational and informational thinking. Journal of Pedagogic Development. Volume 5, Issue 1, March 2015. Retrieved from https://www.beds.ac.uk/jpd/volume-5-issue-1-march-2015/teaching-programming-with-computational-and-informational-thinking.

[21] F. J. García-Peñalvo and J. A. Mendes (2018) Exploring the computational thinking effects in pre-university education. Computers in Human Behavior, 80, 407-411. DOI: http://dx.doi.org/10.1016/j.chb.2017.12.005.

[22] A. Rojas-López and F.J. García-Peñalvo (2016) Relationship of knowledge to learn in programming and evaluation of computational thinking. In Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'16) (Salamanca, Spain, November 2-4, 2016), F.J. García-Peñalvo, Ed. (ICPS: ACM International Conference Proceeding Series, New York, NY, USA: ACM, 2016, pp. 73-77.

[23] A. Rojas-López and F.J. García-Peñalvo (2018) Learning Scenarios for the Subject Methodology of Programming from Evaluating the Computational Thinking of New Students, in IEEE Ibero - American Magazine of Learning Technologies (Volume: 13, Issue: 1, Feb 2018), pp. 30 - 36. Date of Publication: 01 March 2018. Electronic ISSN: 1932-8540. DOI: http://dx.doi.org/10.1109/RITA.2018.2809941. Publisher: IEEE.

[24] Talent Search (2015) Elite: Grade 12+, Institute of IT Professionals South Africa, available at http://www.olympiad.org.za.

[25] UK Bebras Computational Thinking Challenge, answers (2015), University of Oxford, available at http://www.bebras.org.

[26] F.J. García-Peñalvo. 2013. Education in knowledge society: A new PhD programme approach. In Proceedings of the First International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'13) (Salamanca, Spain, November 14-15, 2013), F.J. García-Peñalvo Ed. ACM, New York, NY, USA, 575-577. DOI: http://dx.doi.org/10.1145/2536536.2536624.

[27] F.J. García-Peñalvo. 2014. Formación en la sociedad del conocimiento, un programa de doctorado con una perspectiva interdisciplinar. Education in the Knowledge Society 15, 1, 4-9.

[28] F.J. García-Peñalvo. 2015. Engineering contributions to a Knowledge Society multicultural perspective. IEEE Revista Iberoamericana de Tecnologías del Aprendizaje (IEEE RITA) 10, 1, 17-18. DOI: http://dx.doi.org/10.1109/RITA.2015.2391371.

[29] F. J. García-Peñalvo. 2016. Presentation of the GRIAL research group and its main research lines and projects on March 2016. https://goo.gl/dSZYv7.

[30] GRIAL Group. 2018. GRIAL Research Group Scientific Production Report (2011-2017). Version 2.0. Technical Report Report. GRIAL Research Group, University of Salamanca.