Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

Estrategia didáctica de enseñanza y aprendizaje para programadores de software

Teaching and learning didactic strategy for software programmers

Ensino e aprendizagem de estratégia didática para programadores de software

Elvira Ivone González Jaimes
Universidad Autónoma del Estado de México, México
ivonegj@hotmail.com
https://orcid.org/0000-0002-5328-5586

Asdrúbal López Chau
Universidad Autónoma del Estado de México, México
alchau.uaemex@gmail.com
https://orcid.org/0000-0001-5254-0939

Valentín Trujillo Mora
Universidad Autónoma del Estado de México, México
vtrujillom@uaemex.mx
https://orcid.org/0000-0002-5936-4795

Rafael Rojas Hernández
Universidad Autónoma del Estado de México, México
rrojashe@uaemex.mx
https://orcid.org/0000-0001-6649-067X

Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

Resumen

Introducción. Las estrategias didácticas utilizadas para la enseñanza y el aprendizaje de la programación de *software* son complejas porque se debe adquirir, codificar y recuperar información con base en el pensamiento lógico-matemático para diseñar instrucciones que serán ejecutadas por un ordenador con el fin de resolver problemas de diversa naturaleza. Esto justifica por qué estos contenidos tienen alta demanda en la educación superior de la actualidad, a pesar de que los índices de titulación sean, por el contrario, muy bajos. **Objetivo.** Proponer y evaluar estrategias didácticas generales y específicas para la enseñanza y aprendizaje de programación en *software* con el fin de crear una alternativa para disminuir los registros de bajo rendimiento y la deserción en las carreras que incluyan temas de la mencionada área. **Método.** Diseño cuasiexperiemtal, análisis cuantitativo no paramétrico, longitudinal (cuatro evaluaciones), con una muestra de 78 estudiantes de la carrera de ingeniería en Computación, con distribución al azar en dos grupos experimentales y dos grupos de control en escenarios de aulas inteligentes. **Material.** Programas de *software* con lenguajes C y C++ en plataforma educativa, así como uso de Autogradr y del *Inventario de estrategias metacognitivas*. **Resultados.** Se empleó una prueba de Kruskal-Wallis debido a que se tuvo una muestra sin distribución normal y con escala por jerarquías. Se observó la aceptación de la hipótesis nula en ambas pruebas: 1) H = 0.04, en rendimiento académico superior en los grupos experimentales, la interdependencia en conocimientos es regular alta 0.657, con diferencia de 1.7 en rendimiento promedio, y 2) H = 0.24, en estrategias metacognitivas superior en los grupos experimentales, la interdependencia en estrategias metacognitivas es baja 0.342. Se resalta el área de planificación como fase elemental para la solución de problemas. **Conclusiones.** Se demostró que la estrategia didáctica expuesta en tres bloques específicos de enseñanza y aprendizaje elevó el rendimiento académico y la metacognición de los estudiantes.

Palabras claves: estrategias de aprendizaje, estrategia de enseñanza, programas de computación.

Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

Abstract

Introduction. The didactic strategies used for teaching and learning software programming are complex because information must be acquired, coded and retrieved based on logical-mathematical thinking to design instructions that will be executed by a computer in order to solve problems of diverse nature. This justifies why these contents are in high demand in today's higher education, despite the fact that the degree indices are, on the contrary, very low. **Objective**. Propose and evaluate general and specific didactic strategies for teaching and learning software programming in order to create an alternative to reduce low-performance records and dropout in careers that include subjects from the above area. **Method.** Design quasiexperiemtal, quantitative analysis not parametric, longitudinal (four evaluations), with a sample of 78 students of the career of engineering in Computation, with random distribution in two experimental groups and two groups of control in scenarios of intelligent classrooms. **Material.** Software programs with C and C++ languages in educational platform, as well as the use of Autogradr and the Inventory of metacognitive strategies. **Results**. A Kruskal-Wallis test was used because we had a sample without normal distribution and with scale by hierarchies. The acceptance of the null hypothesis was observed in both tests: 1) H = 0.04, in superior academic performance in the experimental groups, the interdependence in knowledge is regular high 0.657, with difference of 1.7 in average performance, and 2) H = 0.24, in superior metacognitive strategies in the experimental groups, the interdependence in metacognitive strategies is low 0.342. The planning area is highlighted as an elementary phase for the solution of problems. **Conclusions**. It was demonstrated that the didactic strategy exposed in three specific blocks of teaching and learning elevated the academic performance and the metacognition of the students.

Keywords: learning strategies, teaching strategy, computer programs.

Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

Resumo

Introdução As estratégias didáticas utilizadas para o ensino e aprendizagem de programas de software são complexas, pois devem adquirir, codificar e recuperar informações baseadas em raciocínio lógico-matemático para projetar instruções que serão executadas por um computador para solucionar problemas de palavras. natureza diversa. Isso justifica por que esses conteúdos estão em alta demanda no ensino superior de hoje, embora os índices de grau sejam, ao contrário, muito baixos. Objetivo Propor e avaliar estratégias didáticas gerais e específicas para o ensino e aprendizagem de programação de software, a fim de criar uma alternativa para reduzir os registros de baixo desempenho e abandono em carreiras que incluam tópicos da área acima mencionada. Método Projeto Cuasiexperiemtal, análise quantitativa não paramétrica, longitudinal (quatro avaliações), com uma amostra de 78 alunos da carreira de engenharia em Computação, com distribuição aleatória em dois grupos experimentais e dois grupos controle em ambientes de sala de aula inteligente. Material Programas de software com linguagens C e C ++ na plataforma educacional, bem como o uso do Autogradr e do Inventário de Estratégias Metacognitivas. Resultados Um teste de Kruskal-Wallis foi usado porque havia uma amostra sem distribuição normal e escala por hierarquias. A aceitação da hipótese nula foi observada em ambos os testes: 1) H = 0,04, em maior desempenho acadêmico nos grupos experimentais, a interdependência no conhecimento é alta regular 0,657, com diferença de 1,7 no desempenho médio, e 2) H = 0,24 , nas estratégias metacognitivas mais elevadas nos grupos experimentais, a interdependência nas estratégias metacognitivas é baixa 0.342. A área de planejamento é destacada como a fase elementar para resolver problemas. Conclusões Foi demonstrado que a estratégia didática exposta em três blocos específicos de ensino e aprendizagem elevou o desempenho acadêmico e a metacognição dos alunos.

Palavras-chave: estratégias de aprendizagem, estratégia de ensino, programas de computador.

Fecha Recepción: Noviembre 2017 **Fecha Aceptación:** Mayo 2018

Introduction

The didactic strategies used to acquire, encode and retrieve information in software programmers are complex, since they require not only logical-mathematical thinking to formulate mathematical algorithms and designs, but also the ability to retain and manipulate computer language. For this reason, in this study have been selected, exposed and tested teaching strategies to try to facilitate the acquisition of knowledge of software programmers. For this, the previous study of González and López (in press) has been taken as support, where the specific needs that are required to program in C language, specifically in an intelligent classroom scenario (smart classrrom), were observed.

Regarding this aspect, it is worth noting that the activities that are commonly carried out by the software programmer at the professional level are to build instructions that will be executed by a computer to solve various problems of daily life, which is why it is justified why this knowledge has begun to have a high demand in higher education (Kori, Pedaste, Altin, Tõnisson, and Palts, 2016).

The problem, however, focuses on the high levels of dropout and failure during the first years of the race, as shown by the figures from different studies. In effect, according to Kori et al. (2015), in Estonia 32.2% of first-year students abandoned careers related to computer science and information technology during 2015. Likewise, in an inquiry conducted in fifteen Spanish universities linked to the aforementioned area, it was determined that during the first year of the 2015-2016 period the levels of school failure were 22.5% (Universia España, 2017).

In Latin America the results are similar: Lázaro, Callejas, Griol and Durán (2017), for example, carried out a study in Colombia in which it was shown that between 45% and 52% of the students who entered an engineering program in Informatics they had abandoned it. Likewise, and in the specific case of the University of Quindío, Castro, Hernández and Vanegas (2013) reported that in that house of study the students only managed to accredit 27% of the basic subjects, reason for which the dropout rates were in 42%. Following the case of Colombia, but specifically at the Simón Bolívar University in Barranquilla, Azoumana (2013) used data mining to point out that between 2007 and 2012 the percentage of desertion was 65%.

In Argentina, on the other hand, Ferrero and Oloriz (2015) carried out a longitudinal study at the National University of Luján during the period 2000-2010, in which it found a high correlation between the academic performance of students in mathematics and the abandonment of 71% during the first year of engineering careers. Likewise, in Peru Sánchez, Gómez and Gaviria (2016) developed a work at the University of the Amazon, in which they found that in 2012 desertion levels stood at 9%, while in 2012 that figure rose to 20%. In Mexico, finally, the Autonomous University of the State of Mexico (UAEM, 2017) reported that 74.2% of the first-year students of that institution failed the final exams (index that decreased to 30.4% in the fifth year), while the overall terminal efficiency was 51.4%, the degree index was 43.2% and the school dropout rate was 12.35%.

Personal requirements to learn to program

The learning of new C programming languages requires the development of short and long term memory to remember the signs and their respective meanings, which can be achieved in a similar way to how any language is acquired, although adding the practice of logical reasoning -mathematical. For this, of course, you can count on the close support of a tutor, who can prevent the student from continuing to edit errors while in the process of inductive and deductive abstraction, which interrupts the accommodation of information and self-regulation of learning (González and López, in press, González, López, Rojas and Trujillo, in press).

What is going to be learned? Programming software in C and C ++ language

The present study focuses on the development of the programming language in C and its application in C ++, which have several advantages, as indicated below: 1) C languages and their evolution C ++ are powerful and flexible, with orders , operations and library functions that can be used to write most of the programs that run in various computer designs, 2) are used by professionals to develop softwares in most modern computer systems, 3) are used to develop operating systems, compilers, real-time systems and communications applications, 4) they are portable and can be moved from one type of computer to another, and 5) they have a high speed of execution and are almost universal because there are C language libraries and C

++ that support a variety of applications such as graphic databases, text editing, communications, etc. (Joyanes and Zahonero, 2005).

Where you are going to learn (scenario). Smart classroom (smart classrrom)

Learning software programming in a smart classroom or smart classroom has its advantages; for example, 1) increased attention to students and creation of an interactive and collaborative environment (Lozano, 2004, Antona et al., 2011), 2) reflective access to problem solving and implementation of solutions in a language of programming in C (Cairó, 2006), 3) interactivity in the acquisition of knowledge, process of information and transformation of learning in a product (Cook, Augusto and Jakkula, 2009), and 4) at the moment that can be felt and valuing knowledge, allows reflection on what has been achieved, what is useful to propose structured self-dialogues and to evaluate why, how and for what, which stimulates the student's metacognition (Bravo, Hervás, Sánchez and Crespo, 2004) . In addition, there are universal principles of the intelligent classroom that can be used for their generalizable pedagogical configuration (Maheshwari, 2016).

How are you going to learn? Complex knowledge learning process

After considering what and where to learn, it is important to reflect on how you learn, because in this way you can not only understand the way in which the subject knows the object through an intelligent classroom (Sánchez, 2000), but also to know which learning strategies should be used (González and López, in press). In this way, metacognition of complex elements in an intelligent classroom that tries to facilitate and accelerate the acquisition, codification and reproduction of knowledge is promoted through a new educational paradigm (Lozano, 2004). For this, however, a set of skills must be available that are "basically related to the processes linked to the acquisition, organization, retention and use of knowledge" (Gutiérrez, 2005, p.6).

The acquisition is linked to the mental map that the subject possesses and the interrelation with new knowledge, a process in which thought is assimilated, transformed and reorganized until the retention of new concepts is achieved. Then, when the individual returns to the world, adaptation and accommodation arise: "two powerful motors that make the human

being maintain that continuous development of their cognitive structures" (Pinto and Martínez, 1994, p.25).

Metacognition, then, is a cornerstone because it is based on three essential aspects: 1) awareness, that is, the process of conceptualization and manipulation of the object (use of codes and logical processes of the flow of the diagrams), 2) the process of abstraction, which occurs at any stage of development and allows access to new knowledge (to perform simple or complex operations), and 3) self-regulation, where the subject compensates or levels cognitive disturbances, setbacks or discomfort that must face the object in learning (facilitated by electronic devices) (Gonzalez and Lopez, in press).

Selection of teaching strategies as intervention to facilitate knowledge

The concept of didactic strategies "involves the selection of activities and pedagogical practices in different formative moments, methods and resources in the teaching-learning processes" (Velazco and Mosquera, 2015, pp. 1-2). The classification of these are diverse, since different variables must be taken into account; for example, the objective of study, that is, teaching and learning of software programs in C and C ++ languages (Joyanes and Zahonero, 2005). Also, the study scenario, that is, 1) smart classrooms with permanent connection to broadband internet, 2) platforms with educational software designed by experts in C and C ++ programming (Santana-Mancilla, Magaña, Rojas, Niebla and Salazar, 2013; Sharma, 2016), 3) smart lab or smart lab for teaching programming (Alammary, Carbone and Sheard, 2012), and 4) didactic tools such as Autogradr (Soni and Dalal, 2018).

Given the above, in this work we used general teaching and learning strategies (to help acquire all kinds of knowledge) and specific strategies (according to the contextual factor, that is, intelligent classroom, educational platform, intelligent laboratory and Autogradr didactic tool). which are useful in the task of learning the programming language C and C ++) (Legorreta, 2015). Next, each of these is explained:

General strategies

In each of the blocks the teacher proposed a) the objectives and goals, b) the cards of concepts with meaning, laws, rules and principles, c) readings with programming language and d) support groups.

Specific strategies

The first block focused on the intelligent classrooms scenario, which had its pedagogical principle of multiplicity, which at the same time allows the use of various types of resources and stimuli (Maheshwari, 2016). To avoid overstimulation (trigger of distraction) analogy teaching and learning strategies were used, which allow attention to be fixed and anchored to prior knowledge. To establish new knowledge, the coding strategy (nemotechnization) was used as a subsequent process to the acquisition; this implies a deeper and more complex processing because it is integrated to the previous information in structures of broader meaning, which is useful for the elaboration of conceptual maps and flow diagrams (Díaz and Hernández, 2002, Flores and Juarez, 2017).

In the second block, the exploration strategy was used, through which the students were offered reading pages of C and C ++ languages (according to the level of progress, in smart laboratories) and they were motivated to share new texts that they were useful to reinforce the sequence and fluency (Cairó, 2006). Likewise, external resources were used (readings suggested in intelligent laboratories) and internal resources (ways of investigating). From this, the student was able to follow and build didactic sequences (interweaving activities in the laboratory) that favored their significant learning, since it was possible to link what was learned in the classroom with what was researched in the electronic systems (Díaz-Barriga, 2013).

In the third block (where the so-called fragmentation strategy was used) the student was offered a whole to separate it into levels of importance and then in pieces, in this way integration was sought in logical and algorithmic sequences. In other words, exercises of the idiosyncratic fragmentation strategy are first offered, where it starts off completely to be broken down into parts and to analyze the fragments, and then the epigraphic fragmentation strategy is used to reconstruct that whole according to its characteristics. Thus, gaps in learning

are avoided (Joyanes and Zahonero, 2005). Simultaneously, the educational tool for programmers Autogradr was used, which has organization, methodology and schematic resources as a model to elaborate algorithms and execute the design, which favors the bases for the resolution of a problem in programming. This educational tool has a direct impact on the design of programming instructions because it improves the level of cognitive functioning.

On the other hand, and in accordance with the antecedents of the present object of study (learning of the programming language in C and C ++ with use of technology), the theoretical framework used was reconstructivism (based on constructivism), which allows us to add different pedagogical, psychological and social positions that enrich and update it for meaningful learning (Díaz and Hernández, 2002). This, in addition, is in tune with metacognition, an element that plays a major role in meaningful learning because it allows intelligent selection and regulation of learning strategies and techniques, which positively impacts not only motivation and concentration of the proposed tasks, but also in the organization of the time of study, aspects that had been detected as the main cause of the low performance of the students.

Finally, the role of the teacher or tutor (expert and specialist) as knowledgeable of the processes of intellectual development and cognitive abilities in the various stages of student learning should be highlighted.

Background of materials for evaluation

First, the Inventory of Metacognitive Strategies elaborated by O'Neil and Abedi (1996), which was translated and adapted into Spanish by Martínez (2004) to analyze the learning strategies used by university students in the development of the tasks. It focuses metacognitive activities in four dimensions (consciousness, cognitive strategies, planning and control) and relies on the technique of factor analysis, which allows to explain the complex phenomenon of metacognition in a precise and sensitive way, through the validity of constructing the items by their correlation and proven homogeneity (Vallejos, Jaimes, Aguilar y Merino, 2012).

- Technical file: Adults, university academic level, individual and group administration, with an average application time of 20 minutes.
- Test characteristics: Twenty items evaluated with a Likert scale, with five response options (1 = never, 2 = few times, 3 = regular, 4 = many times, 5 = always) and evaluation by hierarchy (low, medium, high and very high) corresponding to the 20, 40, 60 and 80 percentiles.
- Reliability indices α .90 and variance of 40.81, obtained by exploratory factor analysis of maximum likelihood and direct oblimin rotation (Vallejos *et al.*, 2012)

Autogradr, on the other hand, it is an educational tool designed to support teaching and learning in computer science (specifically, for programming) and to create customized programming assignments. In addition, it allows to automatically evaluate the source codes of students and provide instant feedback to students. In this way, the cycle of interaction between instructors and students is shortened. On the other hand, if the source code does not generate the correct answers, Autogradr indicates exactly where the error is located, which helps to debug the programs and to make several attempts to solve the problem (Soni and Dalal, 2018).

Autogradr supports the creation of two types of activities that can be assigned to students. The first is the laboratories, in which the student introduces the source code in the Autogradr editor. The second is the projects, in which the student can upload one or more files with the source code. The laboratories are designed for the practice of isolated topics, while the projects for more complex subjects or that require the application of different concepts. This generates performance indexes in terms of the number of successes obtained in the laboratories or assigned projects (Soni and Dalal, 2018).

Finally, the third type is the educational platform with the subject Language Programming C and C ++, which offers indexes and successes of the activities assigned in a portfolio (Roa, 2015).

Methodology

The present was a quasi-experimental, field, comparative study, with nonparametric quantitative analysis and descriptive and inferential statistics, carried out in four groups (two experimental and two control). The population and sample was the same because we worked with all 78 students enrolled in the first and third semesters of both shifts (morning and evening) of the engineering career in Computing. The assignment of the groups was chosen at random because once the shifts had been selected by lottery.

The two experimental groups had four registers: two longitudinal (weekly: indexes of successes in the educational tool Autogradr, and monthly: index of activities carried out of the portfolio of evidence of the educational platform), and two cross-sectional (biannual: application of the Inventory of strategies metacognitive, and semester: academic performance indexes). In contrast, the two control groups had two cross-sectional registers: one semi-annual (application of the Inventory of metacognitive strategies) and one semester (indexes of academic performance). In four groups the same physical conditions were used (intelligent classroom).

In the qualitative descriptive analysis, 1) the characteristics of the sample were determined, 2) the frequency in percentage an $\geq 80\%$ in activities of the educational tool Autogradr and 3) the portfolio of evidence, in three learning strategies in percentage an $\geq 80\%$.

Likewise, the inferential, nonparametric quantitative analysis, Kruskal-Wallis test, was performed to know the difference between the four independent groups, without normal distribution. In this sense, it has scale by hierarchies of the Inventory of metacognitive strategies and continuous scale in the indices of academic performance (Dawson-Saunders and Trapp, 2002).

Hypothesis

H1: The four groups are identical in terms of academic performance indices.

H0: The four groups are not identical in terms of academic performance indexes.

H2: The four groups are identical in terms of indexes of the Metacognitive Strategies Inventory.

H0 The four groups are not identical in terms of the indexes of the Metacognitive Strategies Inventory.

Decision rule:

If $X^2_c \geq X^2_{tab}$ it is rejected H_1

X_{2tab} with K -1 degrees of freedom (4-1) = 3

Process

- Phase 1: Informed consent. Signature of the letter by students.
- Phase 2: Apply the intervention in two experimental groups (first and third semester). Two theoretical hours with learning strategies, two hours of practice in an intelligent classroom and two hours using the educational tool Autogradr (table 1).

Tabla 1. Estrategias de aprendizaje

Adquirir	Asimilar y transformar Proceso interno	Reproducir	Portafolio de evidencias
Estrategias de atención. Estrategias de exploración.	Estrategias de analogías y codificación, integrando palabras o imágenes. Estrategia de elaboración, propio lenguaje y acciones.	Estrategias de búsqueda de codificaciones (nemotecnia, mapas, matrices, secuencias, etc.)	Evaluación de mapas conceptuales y diagramas de flujo. Evaluación de lecturas en lenguaje de programación.
Estrategias de fragmentación: analizar las partes de un todo organizado; primero, identificar y resaltar lo relevante (idiosincrásico). Segundo, asignar sentido lógico matemático (epigrafiado).	Estrategia de organización: Separa por características o significados (semántica) y luego lo junta en conjuntos con reglas (síntesis), realiza esquemas, secuencias lógicas y secuencias temporales	Generación: Tratar de formular una respuesta por 1) ensayo y error, o libre asociación, 2) ordenación de conceptos, y 3) escribir la respuesta hasta que se ajuste a la imagen reconocida.	Evaluación de la secuencias lógica algorítmica para la solución de problemas en programación.
Estrategias de repetición y corrección: Usar programa Autogradr.	Darse cuenta de aciertos y errores para corrección.	Realización de ejercicios identificando el acierto o error hasta lograr respuestas correctas a la solución de problemas.	Evaluación de actividades de éxito del programa Autogradr

Fuente: Elaboración propia

En grupo de control, dos horas teóricas y cuatro horas de prácticas en aula inteligente.

- Phase 3: Evaluate intervention. 1) Weekly: evaluation of activities of educational tool Autogradr, with a total of 80% of assertiveness and 2) monthly in portfolio of evidence of learning strategies; this includes a) attention strategy, evaluation of conceptual maps and flow diagrams, b) exploration strategy, evaluation of readings in programming

language and c) fragmentation strategy, logical-algorithmic sequence evaluation for problem solving in programming. In each of the strategies a total of 80% assertiveness is required.

- Phase 4: Apply and evaluate. 1) Weekly: activities of educational tool Autogradr and 2) monthly: in portfolio of evidence of the three learning strategies.
- Phase 5: Apply and evaluate. Semiannual: Inventory of metacognitive strategies.
- Phase 6: Collect. Semester: indexes of academic performance in school control.
- Phase 7: Analysis of the data. Qualitative and descriptive analysis of 1) characteristics of the sample, 2) frequency of activities of educational tool Autogradr, 3) portfolio of evidence and 4) indexes of academic performance. Inferential quantitative analysis, Kruskal-Wallis test in the Metacognitive Strategies Inventory and academic performance (Dawson-Saunders y Trapp, 2002).

Results

The descriptive analysis focused on:

1. Characteristics of the sample: Average age = 21.4 years; gender = 85% male and 15% female; shifts = 54% learning C program and 46% learning C ++ program.
2. Weekly evaluation of activities carried out in the educational tool Autogradr. 80% assertiveness was requested to prove (table 2).

Tabla 2. Porcentaje promedio de rendimiento en la herramienta educativa Autogradr

	Grupo experimental 1: aprendizaje programa C n = 42.54 %		Grupo experimental 2: aprendizaje programa C++ n = 36.46 %	
	Laboratorio	Proyecto	Laboratorio	Proyecto
1 a 4 semana	81 %	71 %	83 %	65 %
5 a 8 semana	84 %	79 %	84 %	72 %
9 a 12 semana	85 %	82 %	84 %	79 %
13 a 16 semana	86 %	85 %	82 %	80 %

Fuente: Elaboración propia

3. Monthly evaluations of the evidence portfolio with learning strategies. It included a) attention strategy (evaluation of conceptual maps and flow diagrams), b) exploration strategy (evaluation of readings in programming language) and c) fragmentation strategy (algorithmic logical sequence evaluation for solving problems in programming). In each of the strategies a total of 80% of assertiveness is required to prove (table 3).

Tabla 3. Porcentaje promedio de evidencias en portafolio

	Grupo experimental 1: programa C n = 32.48 %			Grupo experimental 2: programa C++ n = 35.52 %		
	Estrategia de atención	Estrategia de exploración	Estrategia de fragmentación	Estrategia de atención	Estrategia de exploración	Estrategia de fragmentación
1.er mes	84 %	70 %	54 %	82 %	68 %	73 %
2.° mes	76 %	74 %	61 %	78 %	70 %	73 %
3.er mes	81 %	75 %	60 %	83 %	72 %	74 %
4.° mes	86 %	81 %	74 %	85 %	81 %	81 %

Fuente: Elaboración propia

4. Hypothesis tests to know if the four groups are the same or different in a) academic performance and b) in the Inventory of metacognitive strategies. The decision is based on the squared results: if it is greater than or equal to the result that is in the tables, then the true hypothesis is rejected, which means that the four groups are different. Getting in the SPSS program, version 17. Among the four groups, measure academic performance.

Value of the Kruskal-Wallis statistic (H = 0.04) in higher academic performance in the experimental groups, so the true hypothesis is rejected, although the interdependence in knowledge is regular high 0.657 (table 4)

Tabla 4. Estadísticas de contraste

	Rendimiento académico	Interdependencia
chi cuadrada	8.86	0.745
grados de libertad	3	3
sing. asintót	0.04	0.657

Fuente: Elaboración propia

Value of the Kruskal-Wallis statistic (H = 0.24) in higher metacognitive strategies in the experimental groups, so the true hypothesis and its interdependence in metacognitive strategies is rejected, which is also low 0.342 (table 5).

Tabla 5. Estadísticas de contraste

	Estrategias metacognitivas	Interdependencia
chi cuadrada	60.65	0.423
grados de libertad	3	3
sing. asintót	0.24	0.342

Fuente: Elaboración propia

Discussion

In the present study, a didactic methodology was tested that included specific teaching and learning strategies for programmers of software systems as an alternative solution to the low approval rates, reason for which the abandonment of the engineering career in Computer Science has increased. a public university of Mexico.

Now, regarding the limitations of this work, it can be pointed out that both the population and the sample selected were the same, which corresponded to the enrollment of the first and third semesters of the aforementioned career.

On the other hand, in relation to the strengths, the following can be pointed out: the didactic strategy was based on a previous study that allowed to detect the specific needs and the learning strategies that should be used to acquire, codify and reproduce the programming language C in an intelligent classroom scenario (González and López, in press).

Likewise, and because they are powerful and flexible, the C and C ++ programming languages were used, which are frequently used by operating system developers due to their specific use and portability characteristics (Joyanes and Zahonero, 2005). These were executed in intelligent classrooms, which have been designed on pedagogical principles that facilitate and accelerate learning (Bravo, Hervás and Chavira, 2005).

The first learning strategies were based on the need to focus attention on new concepts and then link them with previous records. In this sense, the students were shown strategies of analogies and coding in the classroom (personalizing the computer language) that would help them integrate information and develop conceptual maps and flow diagrams (Díaz and Hernández, 2002). With these strategies the objective set in the activities was achieved: experimental group 1 with 86% and experimental group 2 with 85%.

The second strategy was exploration, through which students were given pages to read the programming language C and C ++ (according to the level of progress), which reinforced the sequence and fluency (Cairó, 2006) . With this strategy, the objective required to fulfill the activities was achieved: experimental group 1 with 81% and experimental group 2 with 81%.

The third strategy was that of fragmentation, which consisted in separating, classifying and integrating logical-algorithmic sequences to try to find the solution to various programming problems. In this way, it was possible to identify the complete processes from entry to exit (Joyanes and Zahonero, 2005). With this, however, the required objective fulfillment of the activities was not achieved: experimental group 1 with 74% and experimental group 2 with 78%.

The fourth strategy (repetition and correction) was based on the use of the Autogradr program, an educational tool designed to support the teaching and learning of computer science, specifically for programming. This is very useful to automatically evaluate and feedback the source codes of students, because Autogradr indicates exactly where the error is for the student to modify it. In this way, it helps to debug the programs to solve the problem.

With this strategy the required objective (fulfillment of the activities) was achieved only in experimental group 1 (laboratory 86% and project 85%) and in experimental group 2 (laboratory 82% and project 80%).

Another of the tools proposed continuously was the reflection, that is, the metacognition before the learning processes and achievements. This was crucial to stimulate self-support and self-regulation, as the students required emotional feedback, an indispensable variable to continue with motivation, since learning in exact sciences is based on high levels of difficulty (Barrón, Zatarain and Hernández , 2014; Flores and Juárez, 2017).

On the other hand, to measure the level in the development of metacognition, the Inventory of metacognitive strategies was used, with which the following dimensions were evaluated: consciousness, cognitive strategies, planning and control (Vallejos et al., 2012).

Regarding the value of the Kruskal-Wallis statistic, H = 0.04, higher academic performance was obtained in the experimental groups, so the true hypothesis is rejected, although the interdependence in knowledge is regular-high 0.657, and the difference in performance It is 1.7 on average.

Finally, in relation to the value of the Kruskal-Wallis statistic, H = 0.24, superior numbers were obtained in the metacognitive strategies in the experimental groups, reason why the true hypothesis is rejected; in addition, its interdependence in metacognitive strategies is low 0.342, and the planning area is highlighted, the elementary phase for solving problems.

Conclusions

For the pedagogical work with C and C ++ programming languages, the detection of learning strategies and needs is required, because in this way an intervention more adjusted to reality can be proposed. In this sense, the didactic strategy developed and tested was as follows: the first block includes learning strategies by analogy and coding to attend and fix knowledge. In the second block strategies are used to explore the reading of software codes, construction of laboratories and projects. In the third block, the fragmentation strategy is taken into account to integrate inputs and outputs in problem solving. All of the above plus the repetition and detection of errors provided by the Autogradr didactic tool.

The general conclusion, therefore, is that the use of the didactic sequence of learning strategies, as well as the Autogradr didactic tool exposed in this study, facilitate the acquisition and implementation of the universal language C and C ++ for software programmer students. This means that academic instruction in this branch of study is indispensable, because

softwares are ubiquitous elements that provide flexibility, intelligence and safety to all complex systems and equipment of daily use, from the automatic lighting of the lights of a house to the complex board of a spaceship.

Therefore, it must be remembered that the adequate teaching of programming languages will reduce the levels of failure and abandonment of future engineers in the first semesters of the career, that is, when learning the basics of universal languages for computing.

References

Alammary, A., Carbone, A. and Sheard, J. (2012). Implementation of a Smart Lab for Teachers of Novice Programmers. ACE '12 *Proceedings of the Fourteenth Australasian Computing Education*, Conference 123, 121–130. Retrieved from dl.acm.org/citation.cfm?id=2483731.

Antona, M., Leonidis, A., Margetis, G., Korozi, M., Ntoa, S. and Stephanidis, C. (2011). A Student-Centric Intelligent Classroom. *International Joint Conference on Ambient Intelligence*, 248–252. Doi: 10.1007/978-3-642-25167-2_33.

Azoumana, K. (2013). Análisis de la deserción estudiantil en la Universidad Simón Bolívar, Facultad Ingeniería de Sistemas, con técnicas de minería de datos. *Pensamiento Americano*, 41-51.

Barrón, M., Zatarain, R. y Hernández, Y. (2014). Tutor inteligente con reconocimiento y manejo de emociones para Matemáticas. *Revista Electrónica de Investigación Educativa*, *16*(3), 88-102. Recuperado de https://redie.uabc.mx/redie/article/view/954/966.

Bravo, J., Hervás, R. and Chavira, G. (2005). Ubiquitous Computing in the Classroom: An Approach through Identification Process. *Journal of Universal Computer Science*, *11*(9), 1494-1504. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.5542&rep=rep1&type=pdf.

Bravo, J., Hervás, R., Sánchez, I. y Crespo, A. U. (2004). Servicios por identificación en el aula ubicua. *VI Simposio Internacional de Informática Educativa (SIIE'04)* (pp. 26-27). Cáceres, España.

Cairó, B. (2006). *Fundamentos de programación. Piensa en C*. Ciudad de México: Pearson Educación.

Castro, J. I., Hernández, D. C. y Vanegas, F. A. (2013). *La deserción escolar en la carrera de ingeniería de Sistemas de la Universidad del Quindío* (trabajo de pregrado). Universidad de Quindío, Facultad de Ingeniería.

Cook, D., Augusto, J. and Jakkula, V. (2009). Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, *5*(4), 277-298. Doi:10.1016/j.pmcj.2009.04.001.

Dawson-Saunders, B. y Trapp, R. (2002). *Bioestadística médica* (3.ª ed.). México: El Manual Moderno.

Díaz, F. y Hernández, G. (2002). *Estrategias docentes para un aprendizaje significativo. Una interpretación constructivista.* México, D. F.: Mc Graw Hill.

Díaz-Barriga, Á. (2013). TIC en el trabajo del aula. Impacto en la planeación didáctica. *Revista Iberoamericana de Educación Superior*, *4*(10), 3-21. Recuperado de http://ries.universia.net/index.php/ries/article/view/340.

González, E. y López, A. (en prensa). Aprendizaje metacognitivo de lenguaje, lógica y matemáticas en *smart classroom*. *Revista Electrónica de Investigación.*

González, E., López, A., Rojas, R. y Trujillo, V. (en prensa).Análisis metacognitivo de competencias adquiridas por tecnologías de la comunicación e información en universitarios. *Revista Innovación Educativa*.

Gutiérrez, F. (2005). *Teorías del proceso cognitivo*. Madrid, España: McGraw-Hill/Interamerican.

Joyanes, L. y Zahonero, I. (2005). *Programación en C. Metodología, algoritmos y estructura de datos* (2.ª ed.). Madrid, España: MacGraw Hill Interamericana.

Kori, K., Pedaste, M., Altin, H., Tõnisson, E. y Palts, T. (2016). Factors That Influence Students' Motivation to Start and to Continue Studying Information Technology in Estonia. *IEEE Transactions on Education*, *59*(4), 255–262. Doi:10.1109/TE.2016.2528889.

Kori, K., Pedaste, M., Tõnisson, E., Palts, T., Altin, H., Rantsus R. y Rüütmann, T. (2015). First-year dropout in ICT studies. *IEEE. Global Engineering Education Conference, EDUCON*, 444-452. Doi: 10.1109/EDUCON.2015.7096008.

Lázaro, A. N., Callejas, Z., Griol, D. y Durán, M. (2017). La deserción estudiantil en educación superior: S. O. S. en carreras de ingeniería Informática. *V CLABES.* Recuperado de http://revistas.utp.ac.pa/index.php/clabes/article/view/1674.

Legorreta, B. (2015). *Estrategias de aprendizaje. Fundamentos teóricos y metodológicos de la educación a distancia.* Universidad Autónoma del Estado de Hidalgo. Recuperado de http://cvonline.uaeh.edu.mx/Cursos/BV/Docentes/pdf/Tema2_estrategias.pdf.

Lozano, A. (2004). El aula inteligente: ¿hacia un nuevo paradigma educativo? [Reseña del libro *El aula inteligente: nuevas perspectivas*]. *Revista Electrónica de Investigación Educativa*, *6*(2). Recuperado de http://redie.uabc.mx/vol6no2/contenido-lozano.html.

Maheshwari, V. (2016). The concept of smart classroom. Retrieved from http://www.vkmaheshwari.com/WP/?p=2352.

Martínez, J. R. (2004). *La concepción de aprendizaje, metacognición y cambio conceptual en estudiantes universitarios de psicología* (tesis doctoral). Universidad Autónoma de Barcelona. Recuperado de www.tesisenred.net/bitstream/handle/10803/2632/Tesis_final.pdf.

O'Neil, H. F. and Abedi, J. (1996). Reliability and validity of a state metacognitive inventory: Potential for alternative assessment. *The Journal of Educational Research*, *89*(4), 234-235. Retrieved from https://doi.org/10.1080/00220671.1996.9941208.

Pinto, S. y Martínez, S. (1994). *La teoría de Jean Piaget y el aprendizaje de las ciencias. Monografía*. Colección Cuadernos del CESU, UNAM. (30), 3-110. Recuperado de https://biblat.unam.mx/es/revista/cuadernos-del-cesu-unam/.

Roa, M. (2015). *Cómo elaborar un portafolio de evidencias.* Recuperado de http://www.itmina.edu.mx/subaca/Portafolio%20de%20evidencias.pdf.

Sánchez, C., Gómez, C. y Gaviria, A. (2016): La deserción estudiantil en el programa de ingeniería de Sistemas de la Universidad de la Amazonia (2012-i a 2015-i): una lectura institucional y antropológica del asunto. *Investigación e Innovación en Ingenierías*, *4*(2), 72-118.

Sánchez, J. (2000). *Nuevas tecnologías de la información y comunicación para la construcción del aprender.* Santiago de Chile: LMA Servicios Gráficos.

Santana-Mancilla, P., Magaña, M., Rojas, J., Nieblas, J. and Salazar, A. (2013). Towards Smart Education: Ambient Intelligence in the Mexican Classrooms. *Procedia. Social and Behavioral Sciences*, (106), 3141–3148. Retrieved from https://doi.org/10.1016/J.SBSPRO.2013.12.363.

Sharma, H. (2016). Effectiveness of EDUCOMP smart classroom teaching on achievement in mathematics at elementary level. International. *Journal of Applied Research, 3*(6), 683–687. Retrieved from http://www.allresearchjournal.com/archives/2016/vol2issue6/PartK/2-6-124-334.pdf.

Soni, T. and Dalal, N. H. (2018). *AutoGradr: Automatically grade programming assignments.* Retrieved from https://autogradr.com.

Universia España (2017). Ingeniería es el área con mayor tasa de abandono escolar en España. *Universia España*. Recuperado de http://noticias.universia.es/ciencia-tecnologia/noticia/2017/06/27/1153624/ingenieria-area-mayor-tasa-abandono-escolar-espana.html.

Universidad Autónoma del Estado de México (UAEM) (2017). *Agenda estadística 2017. México*. Recuperado de http://planeacion.uaemex.mx/docs/AE/2017/AE_2017.pdf

Vallejos, J., Jaimes, C., Aguilar, E. y Merino, M. (2012). Validez, confiabilidad y baremación del inventario de estrategias metacognitivas en estudiantes universitarios. *Revista Psicológica*, *14*(1), 9-20. Recuperado de http://sisbib.unmsm.edu.pe/BVRevistas/rev_psicologia_cv/v14_2012_1/pdf/a02v14n1.pdf.

Velazco, M. y Mosquera, F. (2015). Estrategias didácticas para el aprendizaje colaborativo. *PAIEP*. Recuperado de http://acreditacion.udistrital.edu.co/flexibilidad/estrategias_didacticas_aprendizaje_colaborativo.pdf.

Rol de Contribución	*Autor (es)*
Conceptualización	**Elvira Ivone González Jaimes**
Metodología	**Elvira Ivone González Jaimes**
Software	**Asdrúbal López Chau** **Valentín Trujillo Mora** **Rafael Rojas Hernández**
Validación	**Asdrúbal López Chau** **Rafael Rojas Hernández**
Análisis Formal	**Elvira Ivone González Jaimes** **Asdrúbal López Chau**
Investigación	**Elvira Ivone González Jaimes** **Asdrúbal López Chau**
Recursos	**Materiales de estudio software Universidad Autónoma del Estado de México**
Curación de datos	**Elvira Ivone González Jaimes** **Asdrúbal López Chau**
Escritura - Preparación del borrador original	**Elvira Ivone González Jaimes** **Asdrúbal López Chau**
Escritura - Revisión y edición	**Elvira Ivone González Jaimes** **Asdrúbal López Chau** **Valentín Trujillo Mora** **Rafael Rojas Hernández**
Visualización	**Elvira Ivone González Jaimes** **Asdrúbal López Chau**
Supervisión	**Elvira Ivone González Jaimes**

Administración de Proyectos	**Elvira Ivone González Jaimes**
Adquisición de fondos	**Elvira Ivone González Jaimes** **Asdrúbal López Chau** **Valentín Trujillo Mora** **Rafael Rojas Hernández**