Article **Journal of Technology and Education**
June 2021, Vol.5 No.13 27-31

Programming: From abstraction to practice

Programación: De la abstracción a la practica

TIJERINA-MARTÍNEZ, Felipe†*, TOVAR-GONZÁLEZ, Claudia and GARCÍA-CEDILLO, David Rey

Universidad Tecnológica Santa Catarina, Mexico.

ID 1st Author: *Felipe, Tijerina-Martínez* / **ORC ID:** 0000-0003-0591-228X, **Researcher ID Thomson:** E-6006-2019, **CVU CONACYT ID:** 249445

ID 1st Co-author: *Claudia, Tovar-González* / **ORC ID**: 0000-0001-5785-0792, **Researcher ID Thomson:** E-6006-2019, **CVU CONACYT ID:** 992306

ID 2nd Co-author: *David Rey, García-Cedillo* / **ORC ID**: 0000-0001-5914-7718, **Researcher ID Thomson:** E-6178-2019, **CVU CONACYT ID:** 728974

DOI: 10.35429/JTAE.2021.13.5.27.31 Received: January 25, 2021; Accepted: June 30, 2021

Abstract

This article presents a study based on the experience acquired in 18 years working as a teacher in the Information Technology career in the area of software development. Where it has been seen that when students enter the career the range of knowledge is very varied, since there are students who come from technical high schools in which they have already had experience in the programming area and on the other hand we have students who come from a non-technical high school in which they have no experience in the area. Hence the problem of teaching a programming class in which we can achieve the goal that all students develop their logic. This is why a survey is applied to find out if the student has the basic concepts of the programming area since most students learn to program mechanically, not developing their critical thinking to solve problems other than those solved in class. The contribution of the study shows us that to a certain degree, most of the students have the theoretical concept, but they do not know how to apply it in practice.

Abstraction, Programming, Paradigm

Resumen

Este artículo presenta un estudio basado en la experiencia adquirida en 18 años de trabajo como docente en la carrera de Informática en el área de desarrollo de software. Donde se ha visto que cuando los alumnos ingresan a la carrera la gama de conocimientos es muy variada, ya que hay alumnos que vienen de bachilleratos técnicos en los cuales ya han tenido experiencia en el área de programación y por otro lado tenemos alumnos que vienen de un bachillerato no técnico en el cual no tienen experiencia en el área. De ahí el problema de impartir una clase de programación en la que podamos lograr el objetivo de que todos los alumnos desarrollen su lógica. Es por ello que se aplica una encuesta para saber si el alumno tiene los conceptos básicos del área de programación ya que la mayoría de los alumnos aprenden a programar mecánicamente, no desarrollando su pensamiento crítico para resolver problemas distintos a los resueltos en clase. El aporte del estudio nos muestra que, hasta cierto punto, la mayoría de los alumnos tienen el concepto teórico, pero no saben aplicarlo en la práctica.

Abstracción, Programación, Paradigma

Citation: TIJERINA-MARTÍNEZ, Felipe, TOVAR-GONZÁLEZ, Claudia and GARCÍA-CEDILLO, David Rey. Programming: From abstraction to practice. Journal of Technology and Education. 2021. 5-13:27-31.

* Correspondence of the Author (Email: ftijerina@utsc.edu.mx)
† Researcher contributing as first author.

Introduction

At this time the growth of technology has been by leaps and bounds with respect to Hardware, but unfortunately at the Software level they do not meet the perspectives that are sought, for example, the construction of quality software, generate reusable software and make software with engineering processes.

Always at a historical level, Hardware has been ahead of Software development, so there is a great demand for quality software developers, who exist, but most do not meet expectations.

Various ways to improve are not used them properly to provide the desired solutions as in the case of programming paradigms.

Whenever we are going to solve a problem, we face the difficulty of having to find just that, a solution.

Rarely do we stop thinking there is a structural path to solve any problem, obviously having to go into the minimum of detail depending on the problem. Nevertheless, do we really follow that path or is there a path that will help us to find the right solution?

Seeking to be more practical than abstract that would be the solution to all our projects in the programming area.

Methodology to develop

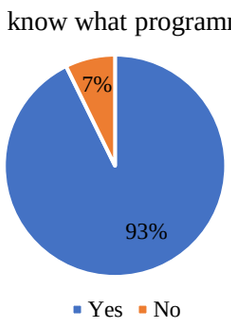
To carry out this research, the Quantitative methodology is used. This questionnaire was applied to 110 people (people who work in the IT area).

Over time, it has been identified that the students enrolled in the different careers of the Technological University have been struggling in the area of programming. In addition, we have found that they used to memorize the processes and not to reasoning the steps of the algorithms that they design and they do it in a mechanical format

A questionnaire is applied; it contains true or false questions. The objective is to verify that 100% of the sample responded positively, because they are people who are already work in the systems area.

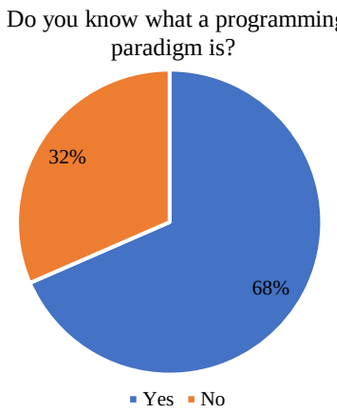
The questions are as follows:

1. Do you know what programming is?



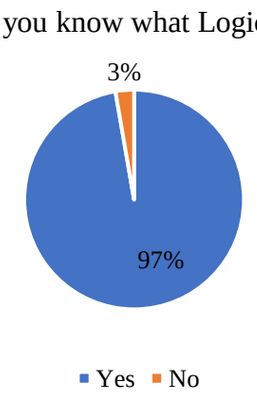
Graphic 1 Question 1

2. Do you know what a programming paradigm is?



Graphic 2 Question 2

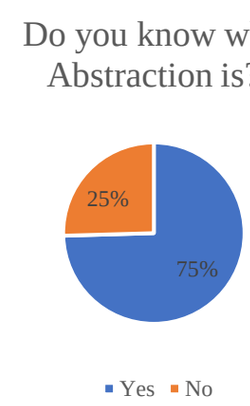
3. Do you know what Logic is?



Graphic 3 Question 3

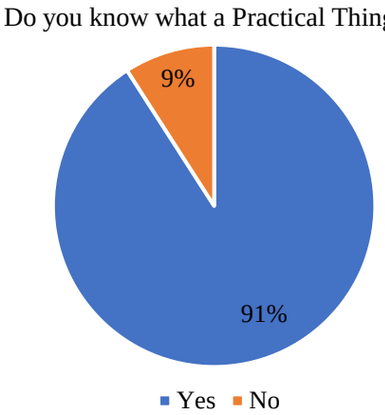
4. Do you know what Abstraction is?

ISSN 2523-0360



Graphic 4 Question 4

5. Do you know what it is something practical?



Graphic 5 Question 5

The answers to the questions are True or False.

The results obtained for each question are shown below.

The information that we get, taking account that the people are from the systems area tell us, we have to analyze the situation urgently. The results obtained were the following:

In the first question, do you know what programming is? 92.7% of the sample answered Yes and No 7.3%. For the question, do you know what a programming paradigm is? 62.7% answered Yes and No, 37.3%. In the question, do you know what is logical? the 97.2% answered Yes and No, 2.8%. In the next question, do you know what abstraction is? the 74.5% answered Yes and No, 25.5%. For the last question, do you know what is practical? the 90.9% answered Yes and No 9.1%

Reviewing the terms, of each concept evaluated in the questions, some descriptions are described below.

Programming. Programming is the process used to devise and order the instructions necessary to carry out a project, prepare certain machines or devices that start working at the time and in the desired way, or develop programs for use in computers.

A programming paradigm is a form or style of software programming.

There are different ways to design a programming language and various ways of working to get the results that programmers require. It is a set of systematic methods applicable at all levels of program design in order to solve computational problems.

Logic is one of the branches of philosophy of an interdisciplinary type, understood as the formal science that studies the principles of proof and valid inference, fallacies, paradoxes and the notion of truth.

Abstraction (from the Latin abstrahere, to move away, subtract, separate) is a mental operation focused on conceptually isolating a specific function or property of an object and that is to think, discarding other properties of the observed object.

As for the practical, we refer to a simple path in which, leads us to a solution as soon as possible.

We are going to start with the practical and ask ourselves a question: What is a paradigm?

The concept of paradigm is somewhat complex, since the use of the term often depends on the area of knowledge from which it is viewed. However, it is generally understood as a synonym for Model.

Throughout history, the different disciplines and aspects of human knowledge have operated according to very different paradigms, that is, to different ways of proceeding and thinking. However, as new discoveries or developments became possible, the human capacity for reason allowed demolishing old structures and building new ones, which translates into a paradigm shift.

ISSN 2523-0360

It is important to break paradigms. We must learn to break concepts (paradigms) to be able to see more options for the development of our activities and not fall into the routine.

The paradigms are divided into two branches, which are the following: Imperative and declarative programming.

The Imperative seeks to define clearly the algorithm that is going to be the source code and the steps that must be executed in an application in a sequential form.

The Declarative describes what the software must solve, that is, the result, not how that Objective was reached.

Discussion and analysis of results

There is a great concern in the area of systems worldwide since there is a shortage of personnel in the programming area of high level.

Lately they have even been inviting well-known personalities in the area of technology (systems development) to come to study programming through different media.

The basis of programming is the logic that is part of mathematics. Taking into account the mathematics as basis, even a single part as Algebra, which helps us to shape our logic and learn how to make decisions. In general, Mathematics is something abstract that we continually take to practicality.

However, what is Abstract? That is which differs from what is not empirically observable but can only be thought or imagined. It is that which can only be conceived in the mind since it has no Material Reality.

Worldwide there is the problem that Students prepare in the area of Programming and programs such as ALICE has been developed whose purpose is to help students of any level not to have difficulty of learning the programming languages since they are the means of communication between humans and Hardware

An abstract problem becomes a concrete problem when the instances and solutions are coded in the form of formal languages.

Abstract problems are usually defined in two parts: the first describes the set of instances and the second describes the expected solution for each instance.

Programming, although it is not a simple subject, and it is not an impossible subject either. The basis of programming is logic, which in simpler words is an achievement of steps to carry out an activity. With very simple exercises and examples, you can understand and exercise logic. Think of things as simple as the steps to change a light bulb, go buy some bread. They are all done with a series of finite consecutive steps.

Once logical thinking is developed, it is very easy to begin by learning the flow of programming, which consists of knowing at which moments in our series of steps there are variables, conditions or cycles. Continuing with the previous examples, a variable can be the amount of money you have to buy bread, which can be a lot or a little. One condition is whether you have money to pay for the bread, in each case there will be a different outcome. For the cycles, suppose that you have a box of five bulbs of which only one works. For this, we have to test all the bulbs to find the right one, every time we test a bulb we are doing a cycle that is repeating the steps repeatedly until it breaks, or rather until we find the bulb that works.

Conclusions

In conclusion, it is determined that students in all areas of knowledge can improve as long as they are taught to handle more practical concepts about their work areas, but in this practical case we are talking about the programming area.

We have been working with special pilot groups with students with disabilities giving real life examples for example, explaining them how the person who sells tacos prepares them, gives you what to drink, and finally charges you and you tell them that this is a multi-user system. Moreover, the student gets involved in the subject and gives ideas about the same concept. In addition, when an example begins to be programmed, they do not have problems and in based on paradigms they locate the solution to the problems.

ISSN 2523-0360

References

Armenta, R. E. (2010). Matemáticas discretas. Cd. de México: Alfa omega.

Boosh, G., & Ccardasi, D. (2013). Orientación a objetos: Teoria y practica. Buenos Aires: Pearson.

Bustamante, O. A. (2007). Introducción a la programación. Cd. de México: Universidad Autonoma de México.

C., M. R. (2012). Código limpio. Madrid: Anaya multimedia.

C., M. R. (2014). Agile software development, principals patterns and practice. San Francisco: Pearson education.

C., M. R. (2018). Arquitectura limpia guia para especialistas en la estructura y el diseño de software. Madrid: Anaya multimedia.

C., M. R. (2019). El limpiador de código código de conducta para programadores profesionales. Madrid: Anaya multimedia.

C:, M. R. (2020). Desarrollo ágil esencial vuelta a las raices. Madrid: Anaya multimedia.

Forero, D. T., Gomez Prado, U. E., & Viola Villamizar, J. B. (2018). Fundamentos de programación. Medellin: Universidad Pontificia Bolivariana.

Mano, M. M., & R. Kime, C. (2005). Fundamentos de diseño lógico y computadoras. Madrid: Pearson.

Regino, E. M. (2015). Logica de programación orientada a objetos. Bogota : ECOE.

Sierra, F. J. (2018). Programación orientada a objetos con C++. Madrid: Editorial RA-MA.
.

ISSN 2523-0360