Experience in Teaching Object-Oriented Programming and Advanced Topics of Programming through the Development of a Video Game Project

Marco Aedo López [1][0000-0001-6608-804X], Elizabeth Vidal Duarte [1][0000-0002-8367-9439] and Eveling Castro Gutiérrez [1][0000-0002-0203-041X]

[1] Universidad Nacional de San Agustín de Arequipa, Calle Santa Catalina Nº117 Arequipa,Perú
{maedol,evidald,ecastro}@unsa.edu.pe

Abstract. This article describes an experience to teach students the fundamentals of object-oriented programming and other advanced programming topics, through the development of a video game, generating a motivating environment for learning them. The object-oriented paradigm and other advanced topics generate important challenges in its teaching that are not easy to deal with, even with a basic knowledge of computer programming. We demonstrate that a teaching supported by the development of a video game project to teach object-oriented programming concepts and other advanced topics is more effective and motivating. Here we describe our experience in the development of this project, to introduce such programming concepts and take into consideration the learning styles of post-millennials or Generation Z. We found that this approach taking into account the styles of visual and kinesthetic learning to teach these topics is more effective and motivating than classic teaching approach.

Keywords: Object-oriented Programming; Advanced Topics of Programming; Motivation in Teaching; Video Games in Learning.

1 Introduction

There are several experiences that show that visual and ludic nature of video games contributes to improving the teaching-learning process of computer programming [1],[2],[3], allowing students to understand abstract concepts that would otherwise be more complex to master.

Fundamentals of Programming 2 (FP2), is a fundamental course of Curriculum 2013 and Curriculum 2017 [4] of the Professional School of Systems Engineering of the Universidad Nacional de San Agustín de Arequipa - Perú, it is centered in teaching of object-oriented programming concepts and other advanced topics, becoming the second programming course in these curricular plans, belongs to the second semester and uses Java as programming language.

Teaching the concepts of object-oriented programming and other advanced topics always had a high degree of complexity, generating a high rate of students who failed the course.

In 2018, techniques based on ludic environments began to be used to teach some basic concepts of object-oriented programming [5], achieving good results, creating a motivating environment for students and awakening in them the desire to master and deepen that knowledge.

It should be considered that the first year students of the professional school who follow the course belong to the so-called Generation Z or post-millennials, pure digital natives, who use technology and electronic devices as a fundamental part of their daily life since they have memory.

Teaching strategies to use with them must be appropriate to their learning style and that is why the development of a video game project was used to achieve a more effective teaching-learning process.

Among the technology they use are video games, a situation that is confirmed in a survey carried out among students for 7 years, the results to the question "do you use video games?" are shown in Table 1.

Table 1. Answers to questioning about the use of Video Games.

Year	Yes	No
2013	90%	10%
2014	92%	8%
2015	95%	5%
2016	97%	3%
2017	97%	3%
2018	96%	4%
2019	98%	2%

In addition, considering the profile of the professional school graduates, which indicates that the graduate will be mainly prepared for the development of quality software, the curriculum offers 3 specializations: Business Information Systems, Systems for Mobile Platforms, and Video Games [4].

During the last years and as part of the same survey, students are asked "what kind of software is most interested in developing in their professional life?", the results are seen in Table 2.

Table 2. Answers to preference of type of software to be developed.

Year	Business Information Systems	Systems for Mobile Platforms	Video Games
2016	20%	25%	55%
2017	20%	26%	54%
2018	21%	24%	55%
2019	20%	24%	56%

As can be seen, the interest is mainly guided to the development of video games, so almost all of the students are users of them and more than half have the intention of directing their professional training to development video games. This generates an intrinsic motivation that will contribute to the mastery of the knowledge to learn and apply in this course.

In this article we present our experience in teaching the fundamental concepts of object-oriented programming and other advanced topics through the development of a strategy video game module, during the development of which the acquired knowledge was applied and consolidated, and students were motivated to more effective learning.

The rest of the article is organized as follows. Section 2 describes the Fundamentals of Programming 2 course. Section 3 shows the learning styles of students in our particular environment. Section 4 shows the design of the video game development project proposal using Java. Section 5 shows the results obtained from students' academic performance, comparing performance when we use the traditional approach to teaching applied in previous years (2013 – 2017) with a ludic approach to teaching applied in 2018 and contrasting it with a ludic approach and framed in the development of a video game project for 2019. Finally, our conclusions are presented.

2 Fundamentals of Programming 2 Course

2.1 Overview

It is constituted in the second programming course in our 2013 and 2017 curriculums, is developed in the second semester, the course lasts seventeen weeks with eight hours per week (two theoretical hours, two practical hours and four laboratory hours) and uses the programming language Java.

2.2 Prior Knowledge

Students who follow this course must have previously approved their prerequisite course: Fundamentals of Programming 1 (FP1), where they learn to program and seek to achieve the specific competencies described in [3].

2.3 Competencies

The 2013 and 2017 curriculums were developed taking into account the student outcomes indicated by the Accreditation Board for Engineering and Technology (ABET) [4]. Where the importance of professional and awareness skills is highlighted, in addition to the development of technical skills to achieve excellence in the training of engineers. Table 3 and Table 4 show the general and specific competencies of FP2.

Table 3. General Competencies of the Fundamentals of Programming 2 Course.

General Competencies
C.c. Responsibly design systems, components or processes, to meet needs within realistic constraints: economic, environmental, social, political, ethical, health, safety, manufacturing and sustainability.
C.p. Flexibly apply the techniques, methods, principles, norms, standards and engineering tools necessary for software construction and information system implementation.

Table 4. Specific Competencies of the Fundamentals of Programming 2 Course.

Specific Competencies
1. Identifies, establishes and integrates the different concepts of Arrays, ArrayList and HashMap, and describes the searching and sorting algorithms, recognizing its importance.
2. Develops, creates and encodes algorithms for the solution of real problems applying the fundamental concepts of Object Orientation, such as own classes, objects, attributes and methods, assessing the importance of the paradigm.
3. Applies, encodes and executes advanced Object Orientation concepts, such as references, overloaded methods and constructors, valuing its use.
4. Applies, encodes, and executes other Object Orientation concepts, such as class and object variables and methods.
5. Analyzes and applies the mechanisms of aggregation, composition, inheritance and polymorphism of the Object Orientation by assessing the power of their use.
6. Conceives the paradigm of event-oriented programming and analyzes the different graphical components that Java provides for the creation of Graphical User Interfaces (GUI) by valuing that paradigm.
7. Assesses the user importance in software development and applies the foundations of Human Computer Interaction (HCI) for GUI design.
8. Conceives the concept of files as a permanent storage medium, valuing its importance.
9. Conceives and applies the Databases as a permanent storage medium, valuing its importance and advantages over files.

2.4 Course Assessment Form

The student performance is assessed in three phases as follows.

Exams. Purely practical where students solve problems applying the theoretical knowledge acquired. They complete three exams throughout the course.

Continuous Assessment. Constituted by individual or group practices, homework, participation in problem-solving activities in the classroom, research work, reading of articles and laboratory assessment (practices and video game project). There are three continuous assessments throughout the course. The final grade is according to Table 5.

Table 5. Weights of Phases Assessments.

Assessment	Exam	Continuous Assessment	Percentage weighting per phase
1st Phase	5%	5%	10%
2nd Phase	20%	5%	25%
3rd Phase	35%	30%	65%
Total	60%	40%	100%

3 Student Learning Style

First year students are part of the so-called Generation Z or post-millennials, individuals who were born during the first decade of the new millennium, and whose philosophy of life and way of learning differ from previous generations.

Applying teaching strategies of the twentieth century to these students is a serious mistake, they no longer tolerate two hours of a master class focused purely on the teacher, the auditory is no longer the main thing for them, instead the visual and kinesthetic is predominant. In [5] we show a study related to the results obtained on the predominant learning styles in our current students.

Thus, it can be synthesized that current students construct their concepts based on digital objects, they quickly absorb information from images and videos, they acquire information simultaneously from various sources (multitasking behavior), expect immediate feedback, collaborative work stimulates them, have short attention spans and value practical activities more than those purely theoretical activities [5].

4 Proposal of the Video Game Project Activities

The activities for the development of the video game project are carried out in the laboratory weekly (two hours for the development of the activity and two hours for the continuous review of the progress of the project). Teams of two members are formed based on affinity, with the option to consult with the members of their team and other teams as well, they are allowed to use the internet, smartphones and other devices, consult with all the sources they want, such as physical and digital books, notes, etc. (all framed in an environment of respect and discipline).

Although these activities take place during class hours, they are also complemented by students outside of that schedule. Almost all students indicate that they meet at other times to continue carrying out the activities of the proposed project and they do so on their own initiative.

Each team has a name established by its members and the video game project is also named by themselves.

Weekly reviews are carried out where students show the performed activities in the last week and the activities to be carried out for the next week, creating a log of such activities. This generates immediate feedback and much needed project follow-up given their understandable inexperience when working on projects. The entire project is carried out in an agile environment, introducing Software Engineering practices and applying HCI principles, thus allowing students to experience software development in their first year [6].

The final presentation takes place in week 16 and 17, where the developed video game is exposed in front of all the other development teams, the course teacher, other teachers and guest gamers (students from other years, teachers or known gamers). They are assigned an exhibition schedule and a specific time for the presentation (ten minutes) and for questions from the public (five minutes), this is recommended but time can be adjusted depending on the number of teams.

Although the grade for the developed project is part of the Continuous Assessment component, what has been learned throughout this activity constitutes milestones for the whole course.

Initially, it is explained to students that they will learn many of the concepts of object-oriented programming and advanced programming topics through the implementation of a module of a basic strategy video game. Their attention is captured, the motivation is automatic given the surprise generated and the expectation of creating their own video game when they are just in the second semester.

A general framework is presented to them and it is explained about the plot of the video game, thus they begin to relate it to some video games that they currently use or used at some time, awakening their imagination.

The proposed plot consists of the fact that the module of the strategy videogame to be developed is framed in the Middle Ages where the nations that currently exist began to be forged, such as England, France, Spain, Germany, etc., those that were born from medieval kingdoms and empires. The module to be developed has to do with the battle part, which is called the tactical aspect of a strategy game. The strategic aspect remains for future experiences and more advanced projects. Battles are fought between 2 armies. Each army is made up of soldiers. Such soldiers have certain characteristics and behaviors that must be implemented. The decision of the winner of the battle must not be arbitrary, it must be based on some metric that the students must propose, so the simplest metric would be "number of army soldiers", which would be fair if all the soldiers were equal, but it would not be the best metric since as the project progresses, different types of soldiers appear, some better than others.

4.1 Proposed Activities for Week 1

Topic. Standard Array.

Objective. Understand and appreciate the use of standard arrays in problem solving.

Statement. Before simulating a battle between two armies, we must consider that each army is composed of soldiers. Given your experience with strategy video games, what soldier data is important? (consider that each soldier will have to be identified individually). Using brainstorming, students indicate that we need to know their name, life level, speed, etc.

Activity 1. Write a program where 5 soldiers are created considering only their name. Input the data and then display it.

Restriction: it will be carried out considering only the knowledge we have of FP1 and without using standard arrays, using only simple variables.

Solution: they created 5 variables of String data type, entered the names and displayed.

Reflection: what if the army consisted of 10, 100 or 1000 soldiers?

It would require more code, creating a very extensive and unmanageable program.

Activity 2. Write a program where 5 soldiers are created considering their name and life level. Input the data and then display it.

Restriction: it will be done considering only the knowledge of FP1.

Solution: they created 5 variables of String data type and another 5 integer variables, entered the name and life level of each one and then displayed them.

Reflection: What if the army consisted of 10, 100 or 1000 soldiers?

It would require more code, creating a long and even unmanageable program.

Activity 3. Write a program where 5 soldiers are created considering only their name. Enter the data and then display it.

Restriction: apply standard arrays.

Solution: they created one array of String data type and size for 5 elements, entered the name of each one, and then displayed them.

Comment: more compact and maintainable code.

Reflection: what if the army consisted of 10, 100 or 1000 soldiers?

No problem, only the array size would have to be changed.

Activity 4. Write a program where 5 soldiers are created considering their name and life level. Enter the data and then display it.

Restriction: apply standard arrays.

Solution: they created 1 array of String data type and size for 5 elements, and another array of integer type and size 5, entered the name and life level of each one, and then displayed them.

Reflection: what if the army consisted of 10, 100 or 1000 soldiers? No problem, only the arrays size would have to be changed.

Activity 5. Write a program where 2 armies are created, each one with a random number of soldiers between 1 and 5, considering only their name. The data is automatically initialized with names such as Soldier0, Soldier1, etc. After creating the 2 armies you must show the data of all the soldiers of both armies and indicate which army would be the winner of the battle.

Restriction: apply standard arrays and methods to initialize armies, show armies, and show the winning army. The metric to apply to indicate the winner is the largest number of soldiers in each army, there may be ties.

Solution: students created 2 arrays of String data type and random size, the names of each soldier were generated, then they showed soldier's data and next the winner was shown.

Comment: more compact and maintainable code.

Deliverable: Video Game v1.0

4.2 Summary of Proposed Activities from Week 2 to Week 17

Due to space constraints, Table 6 shows a summary of the proposed activities.

Table 6. Proposed Project Development Activities from week 2 to week 17.

Week 2 - Topic: ArrayList

Objective: to understand and appreciate the use of ArrayList in troubleshooting as a more flexible alternative to applying standard arrays.

Statement: The solution must be given to Activity 5 from the previous week, but using the ArrayList as the data structure.

Week 3 - Topic: HashMap

Objective: to understand and appreciate the use of HashMap in troubleshooting as a more flexible alternative to applying standard arrays and ArrayList.

Statement: a solution must be given to Activity 1 from the previous week, but applying HashMap as data structure.

Week 4 - Topic: Fundamentals of Object-Oriented Programming 1

Objective: understand and value the creation of classes in solving complex problems.
Statement: For the video game, create the fundamental classes with their attributes and methods. Create your UML class diagram and corresponding Java application.

Week 5 - Topic: Fundamentals of Object-Oriented Programming 2

Objective: understand and value the creation of classes in solving complex problems.
Statement: Create base classes together with their attributes and methods (including constructors, accessor and mutator methods). Create your UML class diagram and corresponding Java application.

Week 6 - Topic: Presentation of the video game project progress

Objective: to publicize the progress made to this point by the team.

Comment: the solutions proposed by each team vary, although each team started from a common base, takes the path it considers correct and justifies their decisions.

Week 7 - Topic: Aggregation and Composition Mechanisms

Objective: understand, evaluate and compare the mechanisms of aggregation and composition.
Statement: now that we have the Soldier class, we can create an army composed of soldiers by applying the mechanism called "composition" or "aggregation" of classes.

Week 8 - Topic: Inheritance Mechanism 1

Objective: understand and assess the class inheritance mechanism.
Statement: in history, armies were composed of different types of soldiers, who had similarities, but also particularities. Apply inheritance to take advantage of it.

Week 9 - Topic: Inheritance Mechanism 2

Objective: apply the class inheritance mechanism for code reuse.
Statement: In addition to the three previously created subclasses, the Soldier superclass can serve as the basis for creating other new types of subclasses. Create new subclasses.

Week 10 - Topic: Inheritance and Polymorphism

Objective: understand and apply the polymorphism mechanism and relate it to inheritance.
Statement: it is possible to create a standard array, ArrayList or HashMap data structure that contains elements that are of different types, it applies inheritance and polymorphism.

Week 11 - Topic: Advanced Topics in Object Orientation

Objective: understand and value advanced mechanisms of object orientation, such as class members, abstract classes and interface classes.

Statement: You can improve your written code by applying advanced mechanisms such as class members (static), abstract classes and interface classes.

Week 12 - Topic: Graphical User Interfaces 1

Objective: to understand and value the use of graphical user interfaces to create more attractive visual programs, applying HCI principles.

Statement: a version of the video game must be created, but in a graphical environment.

Week 13 - Topic: Graphical User Interfaces 2

Objective: to understand and assess the use of graphical user interfaces to create more attractive visual programs using layouts and HCI principles.

Statement: a version of the video game must be created in a graphical environment with basic components, but also using layouts and other components.

Week 14 - Subject: Text Files

Objective: understand and assess the use of permanent storage using text files.

Statement: a version of the video game must be created, but with the option to store the data of a battle in a text file (data from the 2 created armies) and the option to load it later to recreate the battle.

Week 15 - Topic: Binary and Object Files

Objective: understand, assess and compare the use of permanent storage using binary and object files.

Statement: a version of the video game must be created, but with the option of storing a battle in a binary file and another in an object file.

Week 16 y 17 - Final Project Presentation. Present the created product to an audience.

5 Analysis of Results

We must consider that there was always a big problem with the level of approval of the course, the grades were always low during the first five years in which the course was taught. In 2018 we began to carry out small ludic activities that favored academic performance, but after this experience in 2019, the number of approved, general average and standard deviation improved considerably (vigesimal qualification). Data is shown in Table 7 and in Fig. 1.

Table 7. Final Performance Results of the Fundamentals of Programming 2 Course.

	2013	2014	2015	2016	2017	2018	2019
Nº Students	123	112	115	100	115	92	102
Approved	49,15%	40,82%	45,45%	46,00%	52,18%	57,61%	72,55%
Disapproved	50,85%	59,18%	54,55%	54,00%	47,82%	42,39%	27.45%
Average	10,10	9,12	9,21	9,96	10,78	11,58	12,10
Stand. Dev.	3,86	4,36	4,94	3,59	2,65	2,55	2,12
Maximum	17	16	17	17	17	16	16
Minimum	1	1	1	1	2	5	5

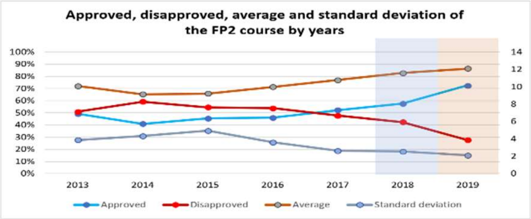

Fig. 1. Annual evolution of the performance level of Fundamentals of Programming 2 course.

6 Conclusions

We presented the laboratory activities that were used to stimulate motivation in our students and teach them, through the development of the video game project, the fundamental concepts of object-oriented programming: class, object, attribute, method, message, composition/aggregation, abstract class, class members, inheritance and polymorphism; and advanced topics such as event-oriented programming, graphical user interfaces and HCI fundamentals; data structures such as standard arrays, ArrayList, HashMap and file usage.

We have shown that teaching these abstract concepts through project development and in a ludic environment allows achieve good results in a generation of students who are pure digital natives. Students learn to teamwork and gain the experience of participating in a software development project.

References

1. Wolz, U., Barnes, T., Parberry, I., Wick, M.: Digital gaming as a vehicle for learning. ACM SIGCSE Bulletin, Vol. 38, No. 1, 394-395 (2006).
2. Sweedyk, E., deLaet, M., Slattery, M. C., Kuffner, J.: Computer games and CS education: why and how. Proceedings of the 36th SIGCSE technical symposium on Computer science education, St. Louis, Missouri, pp. 256-257. USA (2005).
3. Aedo, M., Vidal, E., Castro, E. Paz, A.: Teaching Based on Ludic Environments for the First Session of Computer Programming - Experience with Digital Natives. Revista Iberoamericana de Tecnologias del Aprendizaje, Volume 14, Issue 2, Article number 8736307, 34-42 (2019).
4. Aedo, M., Vidal, E., Castro, E.: Implementation of a Software Engineering Curriculum based on ACM and ABET: A Peruvian experience. 17th Latin American and Caribbean Conference for Engineering and Technology (LACCEI) - International Multi-Conference for Engineering, Education, and Technology. Jamaica (2019).
5. Aedo, M., Vidal, E., Castro, E.: Experience in the teaching of fundamentals of object-oriented programming through the implementation of a strategy videogame. 17th Latin American and Caribbean Conference for Engineering and Technology (LACCEI) - International Multi-Conference for Engineering, Education, and Technology. Jamaica (2019).
6. Sanger, P. A., Ziyatdinova, J.: Project based learning: Real world experiential projects creating the 21st century engineer. International Conference on Interactive Collaborative Learning (ICL). Dubai (2014).