Redesigning CS101: A Learning Based Approach

Alfonso Rodríguez Rivera
Electronic and Computer System Department
ITESM Campus Querétaro
Querétaro, Qro. México. 76130
arodrigz@campus.qro.itesm.mx

Ariel Ortiz Ramírez
Computer Science Department
ITESM Campus Estado de México
Atizapán, Edo. Méx. México. 52926
aortiz@campus.cem.itesm.mx

ABSTRACT

Nowadays, software developers are required to build better systems in less time. We expect computer professionals to work in teams, learn new technologies, solve tough problems, and exceed all past achievements. But as educators, are we really promoting these skills and values? This paper presents the experiences obtained from redesigning an objects-first CS101 course, in which students were made responsible of their own learning practice. Interesting projects developed in teams and built from self-acquired knowledge are the foundation of our proposed scheme.

Keywords

CS101, Java, learning based approach.

1. BACKGROUND

The Monterrey Institute of Technology and Higher Education (ITESM) is a private institution composed by 29 campuses distributed within 26 cities all around Mexico. In 1996, the Institution carried out an extensive consultation involving students, professors, society, and industry. As a result, the 2005 ITESM Mission was developed [2]. This mission has brought about the redesign of the teaching/learning process of all the system's academic programs.

The redesign of the teaching/learning process seeks to develop learning activities that reinforce not only the acquisition of knowledge, but also the achievement of skills, attitudes and values, such as honesty, collaborative work, self-study, problem solving, use and analysis of information, awareness of the country’s social problems, among others.

At Campus Queretaro of the ITESM, we decided to direct the redesign of the CS101 course towards activities that promote and strengthen self-study, collaborative work, and problem solving skills. References [1, 3] served as starting points for the technical content of our redesign.

2. COURSE STRUCTURE

Given this decision, we decided to structure the course activities around projects that would allow students to work using a collaborative approach. We wanted students to develop attractive yet challenging applications. In order to achieve this, we had to replace the traditional lecture-based course approach in favor of a student-activity-based course. And so we did.

Initially, each student has the responsibility of reviewing material and doing exercises, on her or his own, on each of the programming topics required for the projects. We developed an interactive online course material that not only allows reading about different programming concepts, but also allows doing some experimentation. Secondly, the course consists of a two-hour lab session and two one-hour class sessions per week that are aimed at presenting the projects and allowing their development in teams of about eight students.

3. SELF-STUDY MATERIAL

In order to facilitate self-study activities, we developed a set of online learning materials called "Java, Programming with Class". This material may be consulted at the following URL:

http://labcq-master.qro.itesm.mx/~java

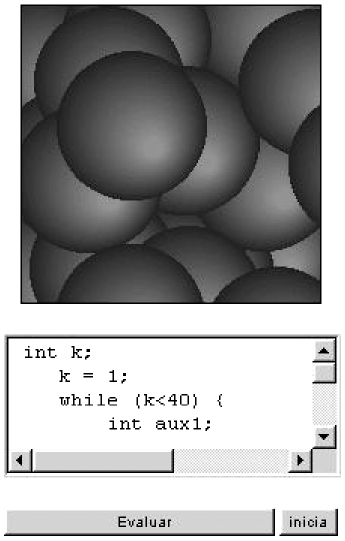
The material is composed of twelve sections. We designed each section so that it could be completely covered in one week. Following is the listing of all sections:

1. Variables and Data Types
2. Object and Classes
3. Functions and Procedures
4. Designing Objects
5. Conditions and Loops
6. More about Conditions and Loops
7. Do-while and For Loops
8. Arrays
9. Array Operations
10. Matrices
11. Vectors
12. Inheritance

Each section contains text and graphics that explain the different elements of the Java language. They also contain interactive activities that allow students to exercise and verify their knowledge. An important element of this material is something we've called “laboratories”, which consist of an interpreter of a subset of the Java language that allows students to write Java code and test their results within the same web pages. These laboratories have been designed to include interactive graphical objects, thus making the material much more appealing to students. Some of the laboratories we developed are:

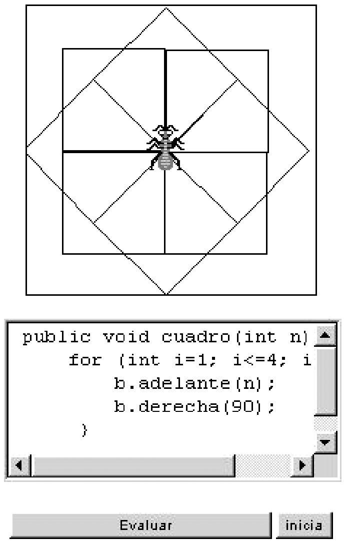
1. Graphic Objects
2. Billiards
3. Bug, a Turtle Graphics environment
4. Arrays and Matrices

Some of theses laboratories are shown in figures 1 and 2.



```
int k;
   k = 1;
   while (k<40) {
       int aux1;
```

Figure 1



```
public void cuadro(int n)
    for (int i=1; i<=4; i
        b.adelante(n);
        b.derecha(90);
    }
```

Figure 2

4. PROJECT BASED LEARNING

One of our main redesign goals was to introduce programming as an interesting and challenging activity. We wanted our students to undertake and solve problems that are traditionally developed in advanced semesters. We wanted to reach further, so we took these projects as the course driving force. All the learning activities rotate around these projects.

During the course, students develop eight projects:

1. "Build your own world", that consists of writing web pages that identify the work groups and each of its members.
2. "Graphical objects", where applications with different graphical interfaces are developed.
3. "Functions", where students develop a complete graphical application that carries out different mathematical computations.
4. "Game", that consists of the development of a graphically animated game that responds to keyboard events.
5. "Interaction", another graphical application. In this case, the application contains different scenes with objects that respond to mouse events,
6. "Multimedia", is a project that includes sounds and animations where different arrays of objects are used.
7. "Image Processing". In this project different applications are developed in which an image is represented as a matrix of Color objects. The operations applied to the image include: change of color, image inversion, and the "impressionist" effect (see figure 3).
8. "Inheritance", the final project that involves the concept of inheritance.

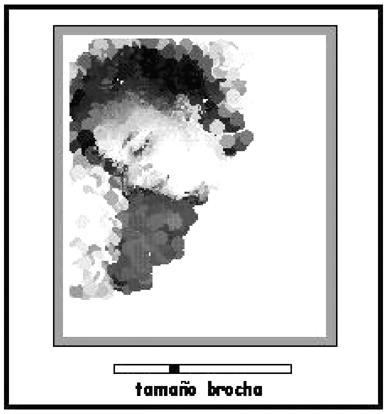


Figure 3

Each project is developed in teams of eight students on average. Afterwards, each student must produce individually a new version of the application.

In order to ease the development of graphical applications and to avoid dedicating too much time to the handling of Java graphics, we created an application framework class library that encapsulates the creation of special objects (such as text fields and buttons) and the event handling routines. The development of an application consists of writing Java code over a skeleton that contains four methods called `start`, `paint`, `respond`, and `mouse` which are responsible for responding to the basic mouse and keyboard events.

5. COLLABORATIVE WORK

In order to promote collaborative work among students, we require projects to be developed in teams. Each team has a minimum of six members and a maximum of ten. Teamwork is done during the class and lab sessions (a total of four hours a week).

For each project, we define the different roles that each student will perform within their teams. One of the roles is the team leader. At the start of every project, every team receives a document that describes the project to undertake. The team members define the different activities that must done in order to complete the project. Every suggestion and commitment done by students is properly documented during each meeting. This documentation is reviewed at the following session.

In order to promote integration among students, the first activity of the semester consists of the development of a web page in which the team is identified with a name, a story and a symbol. This team identity allows us to organize assessments and knowledge evaluation exercises coordinated by students of other teams.

6. RESULTS

Our computer science course has been taught this way since January 1998; every semester we teach an average of five groups composed of 20 students each. Initially, students and faculty had a hard time adapting to this new scheme.

Students faced the challenge of being the first generations using Java as their first programming language. We also expected them to take a different attitude regarding their responsibility towards the learning process. With time, this scheme stopped being a novelty. There are already several generations who have gone through this experience, thus allowing the establishment of help networks among students of different semesters. Furthermore, students have been highly motivated to try to surpass the projects developed by previous generations.

The following text is a quote from a senior year student that began his computer science education using the stated approach:

> “Just a few months away from my graduation, I feel quite confident about the different skills that are required in order to get into a high quality organization. Yet this degree of confidence that I have acquired couldn’t be possible without the personal and professional development process that we, as students, go through at the Monterrey Tech. I can honestly state that the ITESM has provided me two essential skills: self-learning and teamwork. I can assert with gratitude that these two skills were very much encouraged from the very start of my undergraduate education.
>
> “I distinctly remember my first programming course, in which we were responsible of our own learning and knowledge acquisition process. We studied our lessons using a Web platform. This allowed us a lot of flexibility when choosing our studding schedules. It also gave us the opportunity to go deeper into any of the topics we considered interesting. Even though the learning mechanism was centered in the student and not in the instructor (as it has been traditionally), normal classes were very helpful because we had time to ask relevant questions and there was also an important reinforcement of recently learned concepts. Additionally, when developing team-projects, I realized that working with other people requires greater individual responsibility, but with the advantage that one can learn a great deal from the points of view of other people.”

As for the faculty, it was also difficult at first to get them acquainted to their new role as coordinators. Instructors now have to be aware of both teamwork and student self-learning. The professor's task of "teaching" has been limited to a one hour a week question session. This a is quote from a faculty member that teaches a CS101 follow-up:

> “As a CS102 instructor I teach several object-oriented topics using C++. It has been quite interesting to observe those students whose first programming course was taught using the Java programming language. There is a noticeable advantage among those who have this background from those who don’t. Additionally, there are several skills that I’ve been able to perceive in these students: a capacity to learn by their own, a great amount of creativity, and a notorious ability to solve relatively complex problems. This has allowed us to have a much more intense CS102 course, because with the stated skills we can cover much more topics in less time. The class sessions are more dynamic because students are already used to reviewing the class material previous to the class itself.”

In the year 2000, the ITESM System decided to officially adopt the Java language as the first programming language in all its 29 campuses for the following undergraduate programs: Computer Science, Computer Engineering, Information Systems, and Electrical Engineering. The redesigned CS101 course developed at Campus Queretaro was acknowledged by the Institution's Computer Science Academy as the model to follow during the system wide inception of the Java language.

7. GENERAL SUGGESTIONS

The following list of suggestions summarizes what we have learned. We hope that they may be useful to other OO educators.

- Students must understand the "object" and "class" concept right from the very beginning of the course, at the same time they learn about variables. This allows the use of objects when working with conditions, loops, procedures, and arrays. It also allows students to work with object arrays in the same way as they work with integer arrays.
- Pre-built graphical objects allow students to develop simple applications from the beginning that seem quite interesting.
- Instructors should develop online activities that allow students to experiment with programming concepts without requiring the use of a development environment and all its associated time consuming tasks (edit-compile-correct errors-execute).
- The Java libraries for handling graphics, multimedia, Internet connections, and so on, should be used in order to design activities that allow students to develop challenging and interesting applications. The complexity surrounding the use of these libraries may be reduced through the design of new class libraries that hide the "advanced" programming details.

8. ACKNOWLEDGMENTS

The success of this course has been possible thanks to the talent and hard work of the following faculty members at Campus Queretaro: Pedro Pérez, Mayra González, Gabriela Cabrera and Fabiola Uribe. We also recognize the work and dedication of our students.

9. REFERENCES

[1] Decker, Rick and Stuart Hirshfield "programming.java: An Introduction to Programming Using Java". Second Edition. Brooks/Cole. 2000

[2] ITESM System. *Misión del Instituto Tecnológico y de Estudios Superiores de Monterrey*. Centro de Efectividad Institucional del Sistema Tecnológico de Monterrey, 1996.

[3] Stein, Lynn A. "Rethinking CS101 project", at <http://www-cs101.ai.mit.edu/>