Increasing student commitment in introductory programming learning

António José Mendes, Luis Paquete, Amilcar Cardoso
Center for Informatics and Systems
University of Coimbra
Coimbra, Portugal
toze@dei.uc.pt, paquete@dei.uc.pt, amilcar@dei.uc.pt

Anabela Gomes
Engineering Institute of Coimbra
Polytechnic Institute of Coimbra
Coimbra, Portugal
anabela@isec.pt

Abstract— **High failure rates are common in many programming courses worldwide. Many causes for the learning problems have already been identified and different solutions have been proposed. However, the situation remains mostly unchanged. So, new pedagogical approaches are necessary, looking to create learning contexts that motivate students, increase their involvement with course activities, and maximize their learning possibilities. In this paper we present the changes made in the structure of a non-majors introductory programming course, and discuss the results obtained. We also present the results obtained in the first implementation of the new course structure.**

Keywords: learning to program; non-majors; motivation; pedagogy;

I. INTRODUCTION

Introductory programming courses at the Department of Informatics Engineering from the University of Coimbra suffer from high failure and dropout rates as reported in many other high educational institutions [1, 2, 3]. This is a situation that affects mostly novices as those courses are usually placed at the beginning of the curricula [4, 5, 6]. Many authors suggested possible causes for students' difficulties, such as the nature of programming, the students' background and study attitudes as well as the pedagogical strategies commonly used in introductory programming courses [1, 2, 7, 8].

In many higher education institutions, introductory programming courses are offered not only to computer science freshmen, but also to students in other degrees. Consequently, courses normally have many students with different backgrounds, needs and interests. The high number of students that fail or drop out worsens the situation. All these factors prevent teachers from giving individualized attention to each student (or even group of students). The same has happened with the first programming course (called IPRP, a Portuguese acronym for Introduction to Programming and Problem Solving), at our Department. This course is common in Informatics Engineering, Industrial Engineering and Design and Multimedia. As the results were not satisfactory, with high rates of failure and drop outs, the Department decided to make several changes in the teaching structure and strategies, aiming to improve the students' results. This decision was reached after some successful experiments that were made in the three previous years in a small introductory programming course directed to Design and Multimedia Master students, who had very limited or no previous programming experience [9, 10]. The new structure was introduced for the first time in the first semester of the academic year 2011/12.

In this paper we present a detailed view of the changes made and their pedagogical consequences in the context of a non-majors introductory programming course directed to Design and Multimedia students. To assess the results we use official information, like the drop out and failure rates of the last few years including this year and the students' answers to the official pedagogical surveys regularly made by the University. We also use the results of another survey, directed only to students that were repeating the course (they had dropped out or failed in the previous year), which was explicitly focused on the changes made. These students followed the course in the two formats, so their views are important to assess the changes made.

In the next section we present the changes made and the rationale behind them. In the following section we present the results obtained, both in terms of failure rates and the students' opinion. We finalize with some conclusions.

II. WHAT CHANGED

The changes decided at management level were essentially organizational: to create separate courses for students following different degrees and to modify the course class structure. These decisions created the opportunity to make crucial changes at the pedagogical level, especially in the case of the course to Design and Multimedia students (we will call it IPRP-LDM from now on).

The Design and Multimedia degree was offered for the first time in the academic year 2008/2009. Since then, the introductory programming course adopted a weekly 2 hour lecture with all the students together, 3 hours of lab classes in groups of 24 students and 2 hours of free tutorial classes in groups of about 60 students. The new class structure includes only one type of class, with students divided in groups of about 24 students. Each group has 5 hours of classes per week, divided in two different days, one with 2 hours and the other with 3 hours. The same teacher is responsible for all classes of a particular group during the semester. Each class consists of

978-1-4673-1352-0/12/$31.00 ©2012 IEEE

both the introduction of theoretical concepts and practical assignments.

The rationale behind the changes made included a better use of class time, as traditional lectures were not as useful as they should be (class rhythm, examples, and activities were not suited for many students due to the dimension and the very heterogeneous nature of the group). Also, this new class structure would allow the presentation and immediate practice of the different concepts, eliminating problems due to the split between lectures and the different lab groups (even more clear in the case of students who missed lectures and arrived to the lab without a clue about the relevant concepts for that day).

The separation of students following different degrees in separate courses reduced the number of students in IPRP-LDM to 103, 45 freshmen and 58 repeating the course after previous failure or drop out. More importantly, the separation allowed us to shape the course considering the interest most Design and Multimedia students had in design and digital art issues. This was the main reason for choosing the Processing language for the course, instead of Python that was previously used in the joint course. The reason was not language simplicity (on the contrary), but its ability to easily support the development of applications that display drawings, animations and art works. Most course activities had a visual nature, while involving programming concepts common in introductory courses (e.g. selection, repetition, arrays, and an introduction to object oriented programming).

A recurrent question of IPRP-LDM students in previous years was: Why should we learn to program? Many of them did not have any motivation to face the difficulties inherent to learn to program. They felt it was not useful for their professional future. As motivation is very important for student involvement in learning activities the teachers addressed this question in the first class. However, instead of telling them how important programming can be, they asked each student to make a web search about Processing made projects. Each of them had to select a project to present in the next class. The teachers noticed a positive impact in the second class because students wanted to know how those projects were made and also to be able to create their own projects.

The possibility of working with small groups of students and the increase of time availability (5 hours of class per week), allowed the teachers to better know each student, her difficulties, preferences and reactions to teacher interactions. To reinforce the teacher – student relationship we used a less conventional activity in programming courses: students were asked to write about their learning experience every two weeks. Reflections were written in the course Learning Management System (Moodle) and were accessible only to the teacher. This activity allowed teachers to know better each student, as many of them seemed to find easier to write about their problems than to speak about them. It was possible to identify and address some learning issues that were causing difficulties to some students, and also prevent some drop outs through direct interventions with the specific students.

To keep students as committed as possible was one of the teacher's main objectives. To achieve this, some other strategies were used:

- In several assignments students were allowed to include a more creative component. For example, the teacher defined the minimum visual requirements for the assignment (e.g. it has to include a windmill with rotating sails), but the rest of the specification was left open, allowing each student to design and program other components that she/he felt interesting;
- Teachers often tried to encourage students to recognize their efforts and achievements. Errors were always presented as learning opportunities, showing that all programmers make mistakes (including the teacher). It was important to make students conscious of their own progresses, so that they understand that learning is possible if they commit enough;
- Assignments given to a particular student took into consideration her current level as much as possible. This means that in the same class different students could be working on different assignments;
- Students were made aware of their role in learning and the necessary attitude and commitment to be successful. The teachers frequently reminded students about the importance of an active and pro-active attitude towards learning;
- Teachers closely followed students' progress, trying to early detect any difficulties. This allowed teachers to provide early corrective actions, both to small groups sharing similar problems and at an individual level.

Although the changes made in the course structure were essentially organizational, their consequences were mostly pedagogical. There was a better learning context, more suitable to the students' characteristics, and a higher degree of consideration of individual difficulties. All this resulted in more motivation and a deeper student involvement that had a positive impact in their final results, as presented in the next section.

III. THE RESULTS

The results accomplished with the new course structure in the introductory programming course for Design and Multimedia students were measured using three different sources of information: the students' final grades, the results of the university official pedagogical survey, and the results of a questionnaire directed to the students that were repeating the course.

A. Final grades

To evaluate the impact of the innovations introduced in the course, we compared the course final information with the two previous years. The data is presented in Table I. This table shows, for each year, the number of enrolled students, the number of students who followed the entire course (including the final exam), the number of approved students, the rate of assessed students (assessed/registered), the success rate (approved/assessed), the approved rate (approved/registered) and the average of the final marks (expressed in the scale 0 – 20 as used in Portugal).

TABLE I. COMPARATIVE DATA IN THE THREE ANALYSED YEARS

ALL STUDENTS							
	Enrolled students	Assessed students	Approved students	Assessed students rate	Success rate	Approved rate	Final marks Average
2011/12	103	82	59	79.6%	72.0%	57.3%	12.9
2010/11	91	78	18	86.8%	23.1%	19.8%	12.1
2009/10	64	57	17	89.1%	29.9%	26.6%	12.0

It is possible to observe that there was a very positive evolution in most figures, especially in success rate and approved rate. The evolution of the last indicator can be seen in Fig. 1.

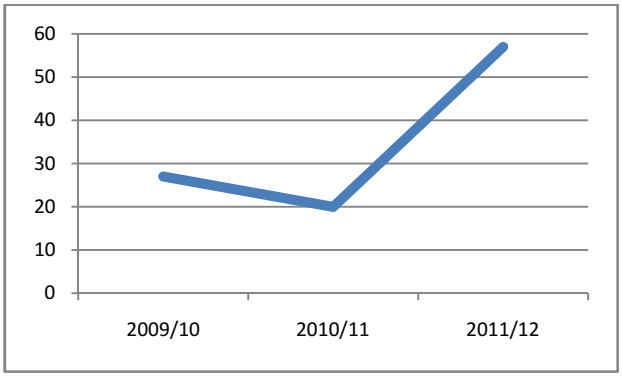

Figure 1. Percentage of approved students

The percentage of approved students increased from 26.6% and 19.8% in the previous years to 57.3% in 2011/12. Considering only the students that were in the final exam, the approved percentage increased from 29.9% and 23.1% to 72%. We performed the same analysis considering freshmen and non-freshmen separately. The results can be consulted in Tables II and III.

TABLE II. COMPARATIVE DATA FOR FRESHMEN IN EACH YEAR

FRESHMEN							
	Enrolled students	Assessed students	Approved students	Assessed students rate	Success rate	Approved rate	Final marks Average
2011/12	45	36	22	80.0%	61.1%	48.9%	13.3
2010/11	48	41	9	85.4%	22.0%	18.8%	12.6
2009/10	38	37	9	97.4%	24.3%	23.7%	12.7

TABLE III. COMPARATIVE DATA FOR NON-FRESHMEN IN EACH YEAR

NON-FRESHMEN							
	Nr of enrollments	Nr of assessments	Nr of approvals	Rate of assessments	Success rate	Rate of approvals	Global Mean
2011/12	58	46	37	79.0%	80.0%	64.0%	12.8
2010/11	43	37	9	86.0%	24.0%	21.0%	11.7
2009/10	26	20	8	77.0%	40.0%	31.0%	11.3

The resulting curves are similar to Fig. 1, although the results were slightly better with non-freshmen than with freshmen (64% approved students versus 48.9%). This is consistent with the idea that learning to program needs time and maturation.

Although average grades also increased, the difference was not so impressive, as it was 12.9 (in a 0-20 scale) this year versus 12.0 and 12.1 in the two previous years (approximately 0.9% increase from previous years). The same analysis separating freshmen and non-freshmen shows that freshmen average grade was higher in the three years.

The percentage of students in the final exam decreased from 89.1% and 86.8% in the two previous years to 79.6% in the last year. This situation causes some concern, as it means that a number of students consistently register for the course, but do not follow it or drop out before concluding it. Further investigation is necessary to understand the reasons for this decrease.

Comparing the percentage of students that got approved in the three years, it is possible to say that the evolution is very positive. The overall approved rate had a sharp increase, supporting the teachers' view that the changes made were benefic for most students.

B. Official pedagogical survey

At the end of the semester the University pedagogical services conducted an anonymous survey about every course offered at the University. The survey included questions about the general conditions of learning (spaces, library, and so on) and particular questions about the student's enrolled courses.

The survey used a Likert type scale with five points (1 to 5), where higher marks mean a better opinion.

The first point to mention is the fact that 90% of IPRP-LDM students answered the survey (a high number when compared with other courses).

The survey consisted of 11 questions about each course followed by the student, such as:

- Appreciate the average quality of learning in the course
- Classify her own learning
- Rate her own participation in learning activities
- Globally classify her own performance in the course

The average marks to each question were between 4.0 and 4.3, reflecting a positive view about the course. Considering that the highest possible mark was 5, this means that most students had a positive view about the course and what they were able to learn.

It is noteworthy to compare IPRP-LDM results in this survey with the global results for the Design and Multimedia degree (considering all courses). In this case the average results in the survey questions went from 3.7 to 3.9. This means that IPRP-LDM was rated above average in all questions included in the survey. This is a very relevant result considering that many other courses could be more appealing to the students, as they are more design and multimedia oriented.

It is also significant to note that in 2010/11 the same survey had already been used, and the average of the student answers about IPRP-LDM were between 2.9 and 3.8, clearly below 2011/12 results.

The University survey also included some questions that allowed the students to give their opinion about the teachers. The results were coherent with the course results, as all averages were between 4.0 and 4.7 for the three course teachers. This means that most students appreciated the efforts the teachers made to promote learning.

C. The Non-Freshmen Questionnaire

To have a deeper insight on the way students reacted to the changes made in the course, we asked the non-freshmen to answer a specific questionnaire, as they had experienced both models. The questionnaire included 10 questions. It was put online and the teachers sent an email to the 58 non-freshmen asking them to give their answers anonymously. Only 30 students answered the questionnaire (17 male and 13 female). Answers were given before students knew their final grades.

The first question asked the students whether they consider the activities in class more adequate or not than those in the previous year. It was a free text answer, but the answer was unanimous. In a way or another, students expressed their agreement with the changes made. They appreciated the use of Processing and considered visual oriented programming activities more interesting and adequate to them. Some also wrote that it was easier to understand the effect of the programming instructions due to the visual impact of the programming language. Finally, some students appreciated the fact that teachers often allowed them to progress at their own pace, giving different exercises to students in different learning stages.

The survey asked students to indicate how often they attended the different types of classes (lectures, labs and free tutorials) in the previous year and classes in the current year. The results are shown in Table IV (in percentage of the total number of classes).

TABLE IV. ATTENDENCE BY CLASS TYPE AND YEAR

	Lectures 2010 - 2011	***Labs 2010 - 2011***	***Free Tutorials 2010 - 2011***	***Classes 2011 - 2012***
[0-25%[	26.67%	0.00%	53.33%	16.67%
[25%-50%[	30.00%	13.33%	30.00%	6.67%
[50%-75%[	30.00%	40.00%	10.00%	16.67%
[75%-100%]	13.33%	46.67%	6.67%	60.00%
Total	100.00%	100.00%	100.00%	100.00%

From the table it is possible to reach some conclusions about these students' class attendance:

- Most of these students did not attend the free tutorials, as more than 80% of them attended less than 50% of those classes.
- More than half of the respondents (56.67%) attended less than 50% of the lectures.
- Most students (86.67%) attended 50% or more labs.

Therefore, it is possible to conclude that the labs were more appealing for the students. As they had difficulties in learning to program, one could expect that they would take advantage of all classes to improve their situation. However, that was not the case, maybe because they did not feel that lectures and free tutorials were useful for them.

On the contrary, in 2011/12 60% of the students attended more than 75% of classes, showing that they felt them useful for their learning.

The questionnaire also included a question to identify drop out situations. The number of students that did not drop out increased from 33.3% in 2010/11 to 96.7% in 2011/12.

For those students who did drop out, the questionnaire included a question about the reasons for that decision. The students mentioned several reasons. The most frequent one was related to failure to achieve the minimum allowed grade in the course project or in some mini-tests that existed in the course evaluation schema. Some students mentioned that they found the course uninteresting, especially lectures. They found them difficult to follow, and useless to clarify doubts, as there was an excessive number of students. The separation between lectures and labs was also mentioned. The teachers at the labs were expecting that the students had understood the concepts and examples presented in the lectures. When that did not happen labs were not very useful, as students could not solve the exercises or even understand the solutions presented by colleagues or the teacher. The time between lectures and labs was also pointed out as a reason for drop out, as some students mentioned it as an added difficulty (when they came to the labs they could hardly remember lectures). Some students simply said that they dropped out because they could not keep up with the course pace.

Analyzing the students' justifications for dropping out in the previous year, we may think that the course structure and activities failed to motivate students to get involved and make

the necessary effort to learn. This lack of commitment is obvious when students mention the time between lectures and labs. Of course, teachers expected some autonomous study in that time, but that simply did not seem to have happened in the case of the respondents.

The questionnaire included a question that asked students to express their opinions about the advantages and disadvantages of the new course structure. Some statements were given and the students had to classify each of them in a five points scale: -2 = "strongly disagree", -1 = "disagree," 0 = "neither agree nor disagree," 1 = "agree" and 2 = "strongly agree". The statements were:

a) There was a better connection between theory and practice.

b) There was a higher proximity between student and teacher.

c) The teacher monitored students more closely.

d) There was more time to clarify doubts.

e) The teaching was more personalized.

f) There was more time to practice programming.

g) It was more tiring due to more practical and intensive work.

h) Time was better organized.

i) The reduction of contact hours (7 hours per week in 2010/11 to 5 hours in 2011/12) was disadvantageous.

j) There was more motivation to program.

The results obtained are presented in Table V. We can conclude that the students had a very positive opinion about the changes. In statements a) to f), h) and j) almost no student expressed disagreement, and even neutral positions were a small minority. This means that for the respondents there were clear advantages in the changes made.

The statement g) got more disperse answers, as 40% of the students were neutral, 46.6% expressed disagreement (saying it wasn't more tiring) and 13.4% expressed agreement. We see this as natural, as the new approach requires a much more active attitude from the students, both in classes and outside classes. Possibly some students see this as tiring, while others like it, since they are learning better than before and think the extra work was worthwhile.

The new model implied a reduction of the number of contact hours. Previously, there were 7h per week (2h lectures, 3h lab and 2h free tutorials), while in the new model the number was reduced to 5h (mostly due to staff constraints, as in the new model all classes were in small groups, creating the need to use more staff hours). Anyway, considering the answers to statement i) 56.7% of the students did not consider this to be a problem, while 33.3% were neutral. So, it seems that, for these students, the changes in the pedagogical strategy compensate the reduction of contact hours.

TABLE V. NEW MODEL ADVANTAGES AND DISADVANTAGES

	-2	-1	0	1	2	∑
a)	3.3%	0.0%	16.7%	43.3%	36.7%	100%
b)	0.0%	0.0%	10.0%	26.7%	63.3%	100%
c)	0.0%	0.0%	16.7%	23.3%	60.0%	100%
d)	0.0%	0.0%	20.0%	36.7%	43.3%	100%
e)	0.0%	0.0%	13.3%	40.0%	46.7%	100%
f)	0.0%	0.0%	16.6%	36.7%	46.7%	100%
g)	23.3%	23.3%	40.0%	6.7%	6.7%	100%
h)	0.0%	0.0%	23.3%	50.0%	26.7%	100%
i)	30.0%	26.7%	33.3%	10.0%	0.0%	100%
j)	3.3%	0.0%	13.3%	36.7%	46.7%	100%

The students were also allowed to mention other advantages and/or disadvantages of the new model. Apart from the issues already covered in the statements a) to j) the students also mentioned as advantages:

- The change to the Processing programming language, as it allowed more interesting activities and is a language used in the creative industries;
- The separation of the Design and Multimedia students from the Informatics Engineering students, as they felt that previously the course was essentially designed for the latter students;
- The students felt more motivated with the learning context, as some of them explicitly mentioned the quality of the teachers and the supportive environment in classes.

As disadvantages, some students mentioned:

- As each group only has one teacher, it is possible that different teachers have different assessment criteria;
- It is possible that different programming issues are presented and practiced differently in the various groups, creating an unbalanced situation when those issues appear in the final exam.

Finally, the questionnaire gave each student the possibility to include any other remark not covered before. Some students used that possibility to reinforce some ideas previously mentioned. Others wrote that this experiment and its pedagogical approach should be extended to other courses in the degree. One student expressed the hope that the better programming knowledge acquired in the course would be helpful in other courses. Some students stressed the pedagogical difference between the current and the previous year.

IV. CONCLUSIONS

This paper describes a pedagogical experiment designed to create better learning conditions for a non-majors introductory programming course.

As the students' results in previous years were far from satisfactory, the Department decided to support a change

proposal made after some experiments in a small size course. There were two main modifications: The separation of students following different degrees in separate courses, and a modification in the class structure. The main idea behind these changes was the creation of a context that improves student motivation and learning support.

We presented the results obtained in the first implementation of the new course structure. To verify the differences we compared the students' results with the two previous years. We also used information obtained from the official pedagogical survey, and from a questionnaire we asked all non-freshmen to answer.

All information available points to the success of this approach in the course. There was a clear increase in the success rate, both considering all students together and freshmen and non-freshmen separately. The official survey brought also positive results, as the students' evaluations on the course were well above the previous year, and also above the average of all courses in the degree. The views of the non-freshmen, that had followed the course in the previous year, were also very positive, stressing the motivational and supportive context created in the course.

The positive evaluation leads us to continue this experiment in the next academic year. The course will have fewer students, due to this year success rate. From the Department point of view this is good news, as less teaching resources will be necessary. The teaching staff is considering some small improvements, but the approach will be similar.

ACKNOWLEDGMENT

The authors would like to thank all students that participated in the experiment.

REFERENCES

[1] Jenkins, T. "On the difficulty of learning to program." In the 3rd Annual LTSN-ICS. 22-27 August 2002. *Proceedings of the 3rd Annual LTSN-ICS*. Loughborough University, United Kingdom, p. 53-58.

[2] Lahtinen, E., Ala-Mutka, K. and Järvinen, H. "A study of difficulties of novice programmers." In the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education. 27-29 June 2005. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*. Monte de Caparica, Portugal, p. 64-68.

[3] Lister, R., Simon, B., Thompson, E., Whalley, J. L. and Prasad, C. September 2006. "Not seeing the forest for the trees: novice programmers and the SOLO taxonomy." *SIGCSE Bulletin*, Vol. 38 (3), pp. 118-122.

[4] Bruce, C. S. and McMahon, C. A. 2002. *Contemporary Developments in Teaching and Learning Introductory Programming: Towards a Research Proposal*. Teaching and Learning Report. Faculty of Information Technology, Queensland University of Technology, Brisbane, Australia.

[5] Dehnadi, S. "Testing programming Aptitude." In the 18th Annual Workshop of the Psychology of Programming Interest Group. 7-8 September 2006. *Proceedings of the 18th Annual Workshop of the Psychology of Programming Interest Group*. Brighton, UK, p. 22-37.

[6] Lister, R. "On blooming first year programming, and its blooming assessment." In the Australasian Conference on Computing Education. 4-6 December 2000. *Proceedings of the Australasian Conference on Computing Education*. Melbourne, Australia. ACM New York, NY, USA, p.158-162.

[7] Gray, W. D., Goldberg, N. C. and Byrnes, S. A. June 2007. "Novices and programming: Merely a difficult subject (why?) or a means to mastering metacognitive skills? [Review of the book Studying the Novice Programmer]". *Journal of Educational Research on Computers*, Vol. 9 (1), pp. 131-140.

[8] Byrne, P. and Lyons, G. September 2001. "The effect of student attributes on success in programming." *SIGCSE Bulletin*, Vol. 33 (3), pp. 49-52.

[9] Martins, S., Mendes, A.J. e Figueiredo, A.D. "Student reflexions as an influence in the dynamics of an introductory programming course." In the 41st Annual Frontiers in Education Conference. 12-15 October de 2011. *Proceedings of the 41st Annual Frontiers in Education Conference*. Rapid City, USA, pp. T1A1-T1A6.

[10] Martins, S., Mendes, A.J. e Figueiredo, A.D. "A strategy to improve students' motivation levels in programming courses." In the 40th Annual Frontiers in Education Conference. 27-30 October de 2010. *Proceedings of the 40th Annual Frontiers in Education Conference*. Washington, USA, pp. F4F1-F4F7.