Informatics in Education, 2018, Vol. 17, No. 1, 45–59
© 2018 *Vilnius University*
DOI: 10.15388/infedu.2018.03

An Assessment of the Teaching-Learning Methodologies Used in the Introductory Programming Courses at a Brazilian University

Edna Dias CANEDO[1*], Giovanni Almeida SANTOS[2], Leticia Lopes LEITE[1]

[1]*Department of Computer Science, Building CIC/EST, Campus Darcy Ribeiro, Asa Norte, University of Brasília (UnB)*
P.O. Box 4466, Brasília-DF, CEP 70910-900, Brazil
[2]*Faculty UnB Gama (FGA), Software Engineering Course, University of Brasília (UnB)*
P.O. Box 8114, Brasília, DF, CEP 72.444-240, Brazil
e-mail: {ednacanedo, giovannix, llleite}@unb.br

Received: December 2017

Abstract. The teaching-learning methodology adopted in the Introduction to Computer Science classes may be a process that makes it difficult to understand the principles of programming language for undergraduate students in Computer Science and related areas, generating high failure and course drop out rates. This paper presents an analysis of the results obtained in the Introduction to Computer Science classes taught in Computer Science and Engineering courses at University of Brasília (UnB). The evaluation questionnaire answered by the undergraduate students in 2017 was analyzed, a validation was performed, and we checked the level of students satisfaction in relation to the evaluated subject and the association among the level of satisfaction, the percentage of practical activities of the discipline, student performance and the level of absenteeism.

Keywords: teaching-learning methodology, programming language, introduction to computer programming, evaluation, failure, drop out.

1. Introduction

New software engineers and casual developers are needed in many different areas. However, students face many difficulties while learning the logic of computer programming, frequently failing in university courses.

[*] Corresponding author: ednacanedo@unb.br; tel.: +55-61-98114-0478

The teaching-learning methodology adopted in the Introduction to Computer Science classes may be a process that makes it difficult to understand the principles of programming language for undergraduate students in Computer Science and related areas, generating high failure and drop out rates.

There are myriads of proposals and research related to the fields of Engineering, Technology, and Education, with the objective of making available resources and contexts that help professors and students make learning more effective. As for the leaning of programming languages, the advances in this research context are easily identifiable, with countless tools and environments presented as proposals for better learning. The works proposed by (Martins *et al.*, 2010), (Edgcomb *et al.*, 2017) and (Bosse and Gerosa, 2017) present some of these tools and problems.

Regardless of the countless resources and tools in existence, at the University of Brasília (UnB), we concluded that a high number of failing students in the basic programming course is a reality for our undergraduate courses.

The results show that it is necessary to seek new teaching-learning proposals to introduce the concepts of programming language for the students of Engineering (Automotive, Aerospace, Energy, Electronic, Electrical, Mechanical, Communication Network, and Mechatronics) and Computing courses, seeking to reduce the number of failures and dropouts.

This paper presents an analysis of the results obtained in the Introduction to Computer Science (ICC), Basic Computing (CB) and Algorithms and Computer Programming (APC) subjects, taught for these courses at University of Brasilia (UnB), between 2008.2 and 2017.1.

Besides that, a survey in the form of a questionnaire is also given to the students of the current semester, 2017.2, regarding the teaching techniques adopted by the professors of the subject, with the objective of identifying improvements in the teaching-learning process.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 presents the analysis of the scenario of the ICC, CB and APC subjects. Section 4 presents the evaluation and analysis of results obtained with the survey given to UnB students. Section 5 presents the conclusions and future studies.

2. Related Work

There are several studies to understand the difficulties faced by computer programming learners. It is known that in the academia there is an intense effort by researchers and professors to try to comprehend the reason behind the difficulty in learning programming concepts. This difficulty of comprehension is an obstacle for an ever higher number of students, as presented by (Jenkins, 2002) and (Lahtinen *et al.*, 2005). Not only that, there is an effort from the leaders of Teaching Institutions and the Government to provide better learning conditions throughout the undergraduate course. Moreover, there has been much debate among computer science professors as to which programming language should be taught to students; for example, adopting an object-first or imperative first approach, as proposed by (Ehlert and Schulte, 2010).

In (Watson and Li, 2014), a systematic review of introductory programming literature is performed, in an effort to statistically consolidate further quantitative evidence on the often cited worldwide high failure rates of programming courses.

The study of (Bennedsen and Caspersen, 2007) was based upon surveying the authors of selected conference papers and performing a statistical analysis of the responses. The study of (Watson and Li, 2014) was based upon performing a systematic review of the literature on introductory programming courses and performing a statistical analysis of the data extracted from relevant articles.

The study by (Hoffbeck *et al.*, 2016) describes the course that was developed to introduce all first-year engineering students to the fundamentals of computer programming within the context of solving engineering problems. The course was designed to utilize active learning techniques by having the students complete a series of laboratory exercises and projects that introduce computer programming and engineering applications. This study describes the origins of the course, the laboratory exercises and projects, how the course was administered, and an assessment of how successful the course was based on student grades, student feedback, and a student survey. The results indicate that the course increased students knowledge of programming in the context of solving engineering problems.

Clearly, teaching effectiveness is a highly complex and very personal process involving a multitude of variables. The study presented by (Galbraith and Merrill, 2012) only attempts to empirically examine the role of one possible factor, the level of faculty research productivity. However, unlike the vast majority of previous empirical studies that simply used student perceptions of teaching, they employed the results of a standardized and quantified student learning outcome assessment process. Few, if any, empirical studies exist that utilize this type of school-wide standardized student outcome measurement for teaching effectiveness.

In (Murphy *et al.*, 2017) there is a report with the results of the first survey of introductory programming courses (Number of courses = 80) taught at United Kingdom (UK) universities as part of their first year computer science (or related) degree programs, conducted in the first half of 2016. It is a report on student numbers, programming paradigm, programming languages and environment/tools used, as well as the underpinning rationale for these choices. The results in this first UK survey indicate a dominance of Java at a time when universities are still generally teaching students who are new to programming (and computer science), despite the fact that Python is perceived, by the same respondents, to be both easier to teach as well as to learn.

This survey presented by (Murphy *et al.,* 2017) provides a starting point for valuable pedagogic baseline data for the analysis of the art, science and engineering of programming, in the context of a substantial computer science curriculum reform in UK schools, as well as increasing scrutiny of teaching excellence and graduate employability for UK universities.

In the governmental level, programs with social and digital inclusion policies are offered, both for college and high school students. Other investments include renovating the infrastructure of Universities, Federal Institutes, and Federal and Local Public Schools, improving the technology and equipment of the classrooms and laboratories, along with several programs to encourage teacher qualification. Even with the amount of programs and research, there are still problems in learning programming.

2.1. *Learning Programming Languages*

At the UnB, there's an agreement between professors and researchers involved with the Intro to Programming subject that learning to program is not a trivial activity, since it introduces a series of cognitive requirements to the daily life of the student, going further than technical requirements.

In their majority, these cognitive requirements incorporate a need for the student to change their way of thinking and acting in their academic life to a different reality than the one they got used to during high school.

If we consider that they're met with this different way of thinking in the first semester of their course, changes need to be incorporated/absorbed in a short period of time, since the subject lasts for a semester. The subject load is of 60 hours of class that usually begin in March and finish in the end of June, or begin in August and finish in the beginning of December of each year.

The cognitive requirements pointed out by (Wilson and Shrock, 2001) and (Wiedenbeck, 2005) are:

1) The resolution of problems is a competency that involves cognitive processes such as creativity and rationality, through a set of mental meta-skills that sometimes go unnoticed, and which are supported by other skills, such as reading and interpreting the description of a problem.
2) The full understanding of the requirements of a programming paradigm is not a trivial activity, and entails an inherent degree of difficulty. This understanding involves abstraction and problem-solving skills.

A majority of professors of programming subjects try to make the student understand that to program is, first and foremost, an exercise of basic reasoning skills (reading, writing, and calculating) and mental skills (comparing, describing, interpreting, classifying, and analyzing) that develop over constant practice and exercise. It is necessary to put in a considerable amount of time to pass a programming subject, especially in the first semesters of an undergraduate course.

3. Selected Analysis of the ICC, CB and APC Subjects

Teaching evaluation is described as the process whereby the quality of teaching is assessed. This measure of assessment can be conducted using formative and/or summative approaches. In higher education, formative assessments of teaching are focused on providing instructors/professors with information that can help them to improve their teaching, as presented by (Pitterson *et al.,* 2016).

3.1. *Programming Subjects at the UnB*

Several programming subjects are offered in the UnB courses. The first contact for the student comes with the Intro to Computer Science (ICC) or Basic Computing (CB) subject, which will be called Algorithms and Computer Programming (APC) starting 2015.2, even though it will maintain the syllabus and workload.

The subject program for the Technology/Engineering courses offered by UnB has a set of differentiated subjects, but all programs offer the students plenty of subjects of programming languages, besides the first ones (ICC, CB and APC). It is critical that the student has a good understanding and comprehension throughout this first contact with programming subjects. As such, understanding the teaching/learning process of the first programming subject can help professors and students in the task of developing a more effective learning environment, this minimizing the difficulties found by students in such subjects. This is the main contribution hoped to be achieved with this study.

Fig. 1 presents an overview for the enrolled, passing, and failing students in the Intro to Computer Science subject in Engineering courses at UnB. In the second semester of 2008 (2008/2), we had 251 enrolled students. In the first semester of 2013 (2013/1), we had 408 students enrolled in the subject.

As can be seen in Fig. 1, in the second semester of 2008, the students a failure rate of 25% (2008/2). It can be seen that this rate went up with each passing term. In 2010/2, for example, the failure rate was at 52%. In 2012/2, the failure rate was of 58%. In the term in which the ICC subject was offered, 2013/1, this rate was at 54%.

Starting from 2013/2, the Intro to Computer Science was renamed to Basic Computing in all five Engineering Courses, namely: Aerospace Engineering, Automotive Engineering, Energy Engineering, Electronic Engineering, and Software Engineering.

Fig. 2 presents the number of failing and passing students in the second term of 2008 (2008/2) in the Computer Science, Computing, Electrical Engineering, Communication Network Engineering and Mechatronics Engineering courses. The Computing course had the highest student failure rate at 43%. The course with the lowest failure rate this semester was Mechatronics Engineering at 4%.

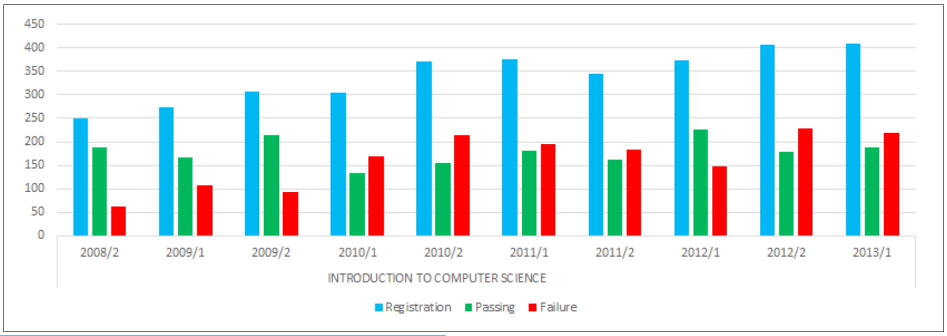

Fig. 1. Registration, Failing and Passing Students in the Introduction to Computer Science.

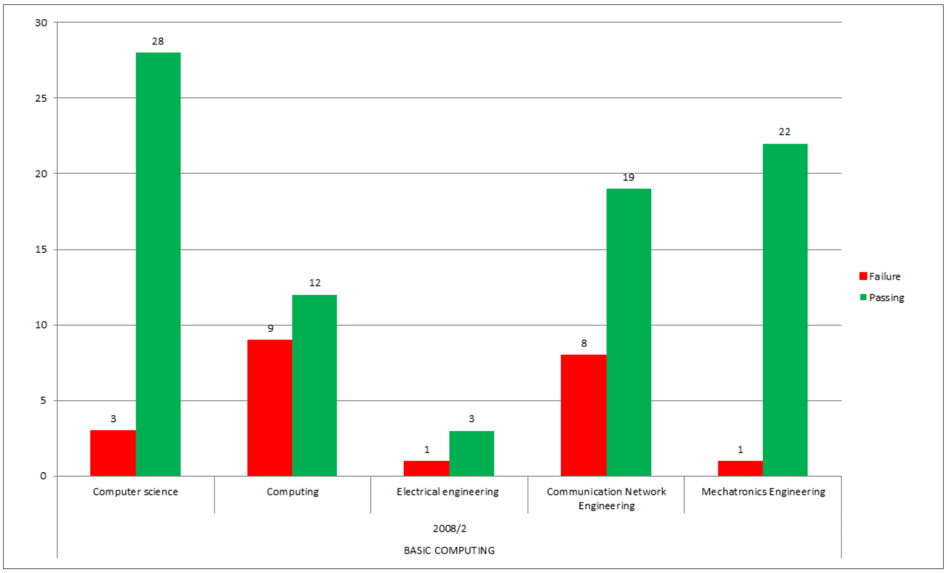

Fig. 2. Failing and Passing Students in the Computer Science, Computing, Electrical Engineering, Communication Network Engineering and Mechatronics Engineering Courses.

Fig. 3 presents an overview of the second semester of 2013 (2013/2) in all Engineering and Technology courses at UnB. In the Fig. 3, it can be observed that the Automotive Engineering course had the highest failure rate at 63%, followed by the Electronic Engineering course at 53%. The Computer science course had a failure rate at 46% and the Mechatronics Engineering course had a failure rate at 44%. The Computer Engineering course had a failure rate at 32%. The course with the lowest failure rate was Communication Networks Engineering at 20%, followed by the Software Engineering course at 19%.

Fig. 4 presents the scenario obtained in the first semester of 2017 in Engineering and Technology courses at UnB. In the Figure, it is possible to identify that the Automotive Engineering course once again had the highest failure rate, 36%, followed by the Aerospace Engineering Course at 27%. The courses with the lowest failure rate were Computer Science, at 13%, and Software Engineering, at 14%.

The Brazilian framework established by the federal government is the National Exam for Student Performance (ENADE), a component of the National System for the Evaluation of Higher Education (SINAES) which, besides cognitive contents, also evaluates the infrastructure and opens up space with some questions about the teaching-learning process (Garrido *et al.,* 2017). Since it is a tool that deals with the big picture, in order to deal with the courses of various natures, it brings information with little specificity for the evaluated Institutions. Even though there is a series of questionnaires available for the evaluation of teaching, these tools prioritize the acquisition of global data on teaching in these institutions, not focusing in the satisfaction of the student in relation to the general and specific aspects of the subjects offered throughout their education.

The students satisfaction with a given subject, when reviewed in literature, regards the subject as a whole, exploring aspects such as the infrastructure, the professor, teaching strategies and methodologies, professor evaluation, different ways of grading in tests and assignments, breaching psychological and skill aspects, as reported by (Cohen, 1981), (Al-Jishi *et al.*, 2009) and (Kahneman *et al.*, 1993).

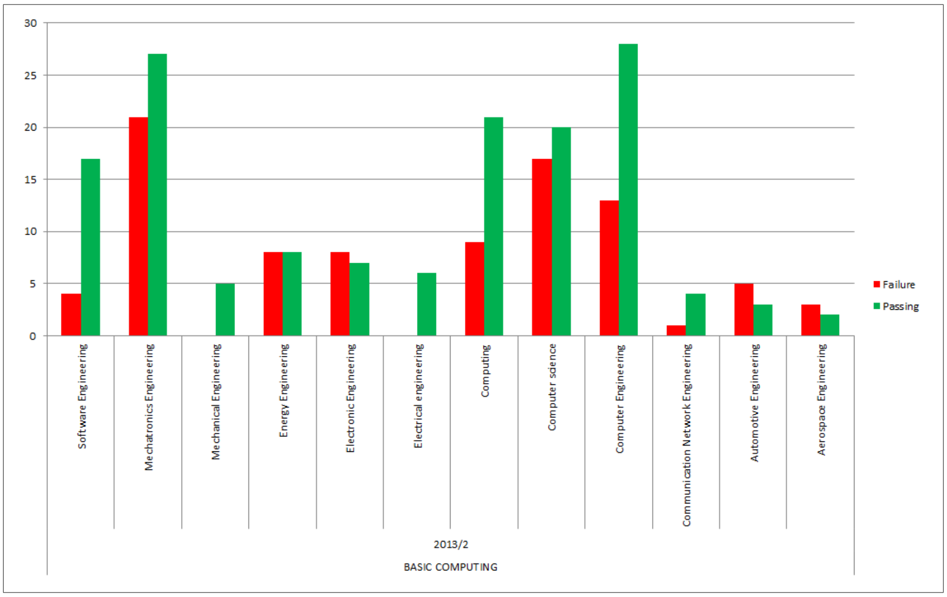

Fig. 3. Failing and passing students in the Technology and Engineering courses in 2013/2.

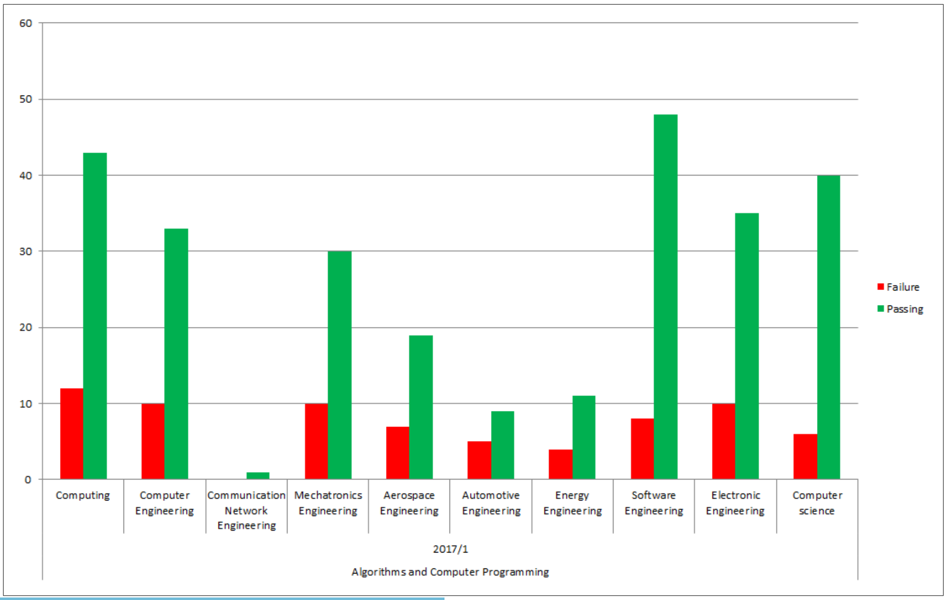

Fig. 4. Failing and Passing students in Engineering and Technology Courses in 2017/1.

4. Analysis of Results

The evaluation tool came to be thanks to the interest by some UnB professors, in 2017, in knowing the level of satisfaction of UnB's students in relation to the introductory programming subject and in the potential of information that could be raised in a student-focused subject evaluation.

A group of professors of UnB's Software Engineering and Computer Science course developed an internal tool to evaluate these subjects. The main advantage of creating ones own questionnaires is that they can correspond to the college's traits and necessities. After many literature reviews and group discussions, the first version of the tool was tested and then made available for all students of engineering and technology courses.

The tool is a questionnaire (https://canedo.typeform.com/to/K2OrV5) that seeks to assess the satisfaction of the student in relation to the different domains of the subject. This includes an overview of the subject and various specific aspects of it, in a quantitative way with open space for a qualitative expression. Thereby, as they evaluate the subject, the student actively participates in identifying the problem related to the high level of failure and high dropout rates and, especially, in the discussion of possible improvements. This allows the student to acquire ways of analyzing their action in a more critical manner, making them take a role of responsibility in the teaching-learning process. (Moreno *et al.,* 2007). To the university, it allows for the diagnosis of the status of the discipline in the Engineering and Technology courses at the UnB, detecting its potential weaknesses and strengths, problems, and successes.

At the launch of the questionnaire, there was considerable resistance by the faculty. Many felt threatened, as they believed it would focus on investigating and punishing them for their teaching roles; they believed the students wouldn't be competent enough to evaluate their teaching methodology, and so on.

4.1. *Results*

It is known that the student passing rate for a certain subject does not necessarily mean that the teaching methodology or computational resources employed by the professor are successful. Therefore, how to evaluate the success and benefits achieved in student learning thanks to a certain teaching methodology?

As a way to evaluate the results achieved with the methodology adopted by the professors of ICC, CB, and APC, some research questions were defined to compose an investigative and evaluative questionnaire. This questionnaire was made available to all students who had already taken CB, ICC or APC classes.

In total, **637** students answered the questionnaire. The obtained results were as follows:

4.1.1. *In Relation to the Subject Taken ICC/CB/APC*

Of the total number of students who answered the questionnaire, 21% took ICC, 43% took CB, and 36% took APC. This is the second semester in which it is offered in the

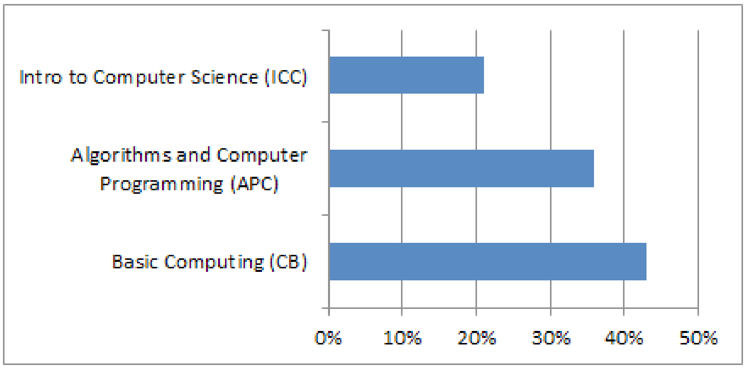

Fig. 5. Subject taken by five Engineering students.

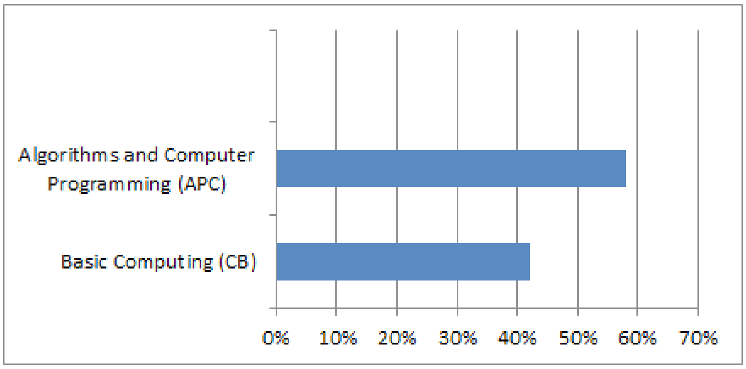

Fig. 6. Algorithms and Computer Programming and Basic Computing Subject taken by students.

five undergraduate courses, namely: Aerospace Engineering, Automotive Engineering, Energy Engineering, Electronic Engineering, and Software Engineering. Fig. 5 represents this result.

Fig. 6 presents the same result for the other courses of the Computational Science Department and the Faculty of Technology in the UnB. About 58% of the students took the Algorithms and Computer Programming subject, and 42% took the Basic Computing subject.

4.1.2. *In Relation to Finding the Subject Challenging, Considering Learning and Passing Rates*

Fig. 7 presents the results of the survey. 11% of the students considered the subject a little challenging. 62% considered it to be one of the most challenging, and 27% considered it very challenging. Without a doubt, the subject is challenging for most students, since some of them don't grasp its real necessity for the course.

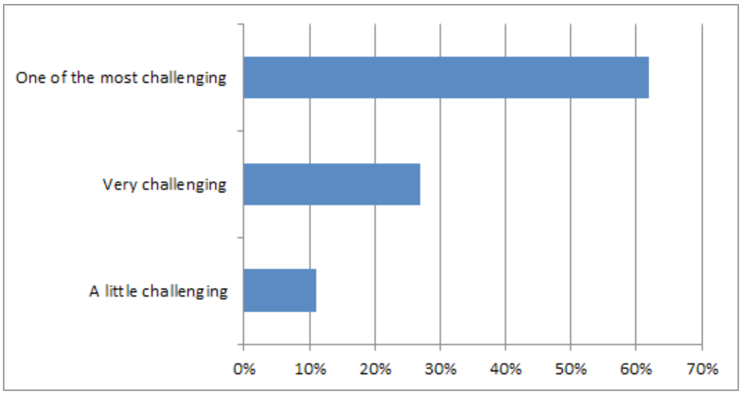

Fig. 7. Students' Answers to the Question: Did You Consider the Subject Challenging, in Regards to Learning and Passing Rates?

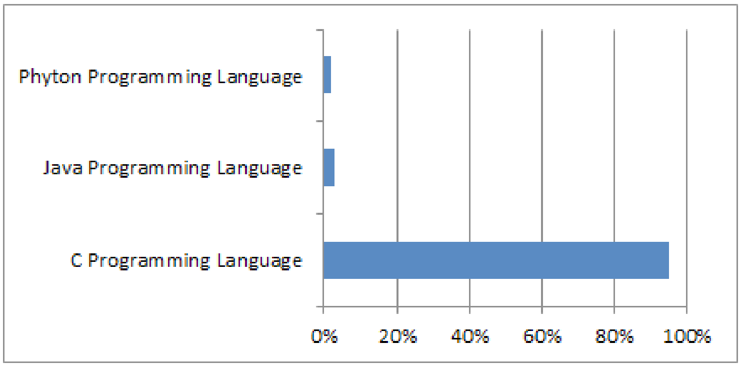

Fig. 8. What programming language is used in class?

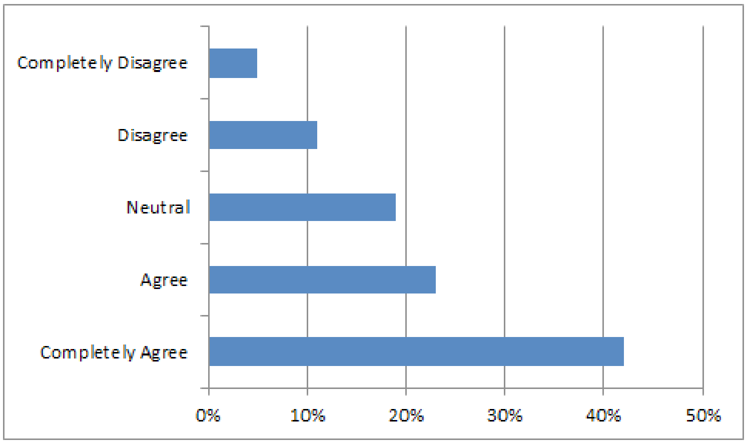

Fig. 9. Opinion of the students in regards to the methodology of the professor assigned to the subjects.

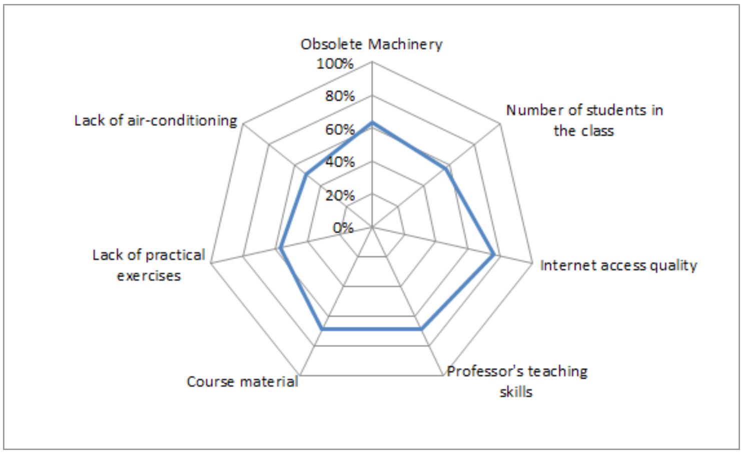

Fig. 10. Problems that negatively impact learning.

4.1.3. *What Programming Language is Used?*

Fig. 8 presents the results obtained in relation to which programming language was taught when the student took Intro to Programming. About 95% of the students learned the C programming language, 3% learned Java, and 2% learned Phyton.

4.1.4. *In Regards to the Teaching Methodology of the Professors Assigned to the Subjects*

Fig. 9 presents the results of student satisfaction in relation to the methodology adopted by the professor in the subject. 23% agree that the professor assigned to either subject has good teaching methodologies. 42% completely agree to the statement, 19% were neutral, and 11% completely disagree with it. 5% disagree that the professor had good teaching methodologies.

4.1.5. *Listed Problems as Being Negatively Impactful in Students Learning*

Fig. 10 presents the results related to this question. Within the listed problems, the ones with the highest negative impact mentioned the students are: the professors methodological aspects at 69% and the amount of students in the classes at 57%. Currently, classes have between 40 and 60 students. 76% claim that internet access quality is a problem. 63% mentioned the obsolete machines found in the laboratories, while 69% blame the course material made available by the professors, and 57% mentioned the lack of practical exercises. Lack of air-conditioning at 51%.

4.1.6. *In your Opinion, is the Infrastructure (Laboratories) Provided by the UnB a Good Environment to Take the Subjects?*

Fig. 11 presents the obtained results about the infrastructure (laboratories) at UnB. About 18% completely agree, 45% agree, and 18% are neutral. Meanwhile, 16% disagree and 3% completely disagree that the infrastructure offered by the FGA is an adequate environment for the learning of the subject.

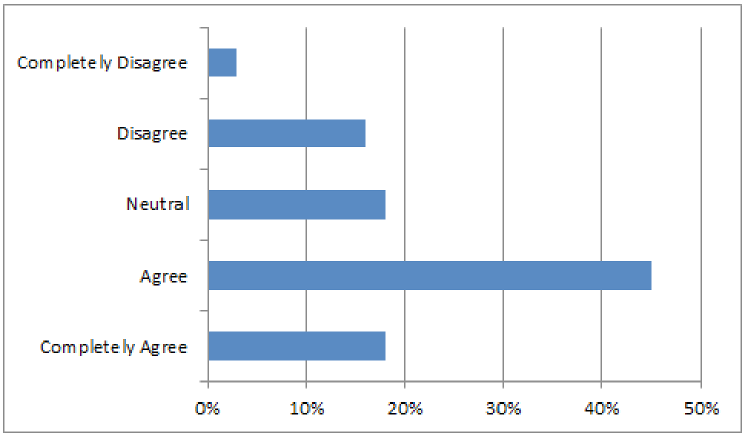

Fig. 11. The Infrastructure (Laboratories) Provided by the UnB a Good Environment to Take the Subjects.

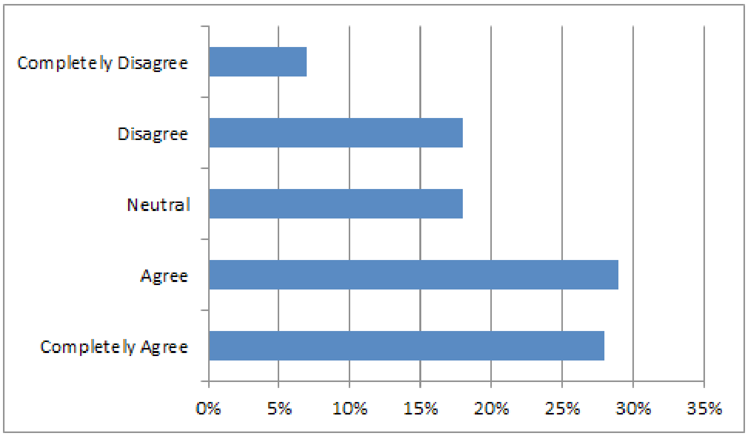

Fig. 12. The Support by the Assistants is Enough to Learn the Subject.

4.1.7. *The Support by the Assistants Enough to Learn the Subject?*

About 28% of the students completely agree and 29% agree that there is enough support. 18% are neutral about it. 7% completely disagree and 18% disagree that the support by assistants is enough to learn the subject. Fig. 12 shows this scenario.

4.1.8. *Is the Teaching Practice Adopted by the Professors Adequate to the Level of Difficulty of the Subject?*

In regards to the teaching practice adopted by the professors being adequate to the level of difficulty of the subject, 30% of the students completely agree and 31% agree. 14% of the students are neutral. 7% of the students completely disagree, and 18% disagree that the teaching practice is adequate. Fig. 13 presents the results related to practice adopted by the professors.

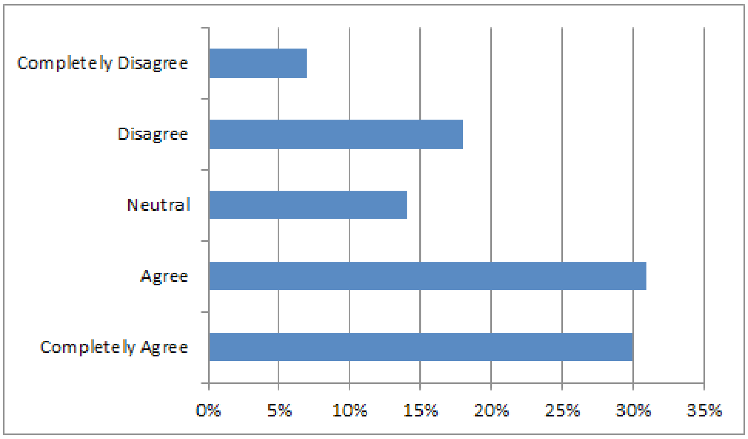

Fig. 13. The teaching practice adopted by the professors is adequate to the level of difficulty of the subject.

5. Conclusion

Analyzing the information from the database of UnB's academic system, it is possible to see a high rate of failure in the ICC, CB and APC subjects.

The questionnaire allowed us to understand a bit more of this situation thanks to the students answers.

The students considered the UnB infrastructure a negative factor that needs to be remedied. There are many broken machines in the laboratories, which calls for an investment for better lab equipment. It is also necessary to rethink the number of students in each class. A reduced number of students would allow for a closer and more personal teaching practice by the professor and better, more efficient understanding by the student. Besides, the material made available by the professors needs to be improved, adding more practice exercises and their respective solutions.

Within the currently adopted practices, we can identify the ones that are well received by students, such as:

1) The inclusion of assistants: 71% of the students claim they look for the assistants to look for help with the subject.
2) Perception of the importance of the subject: 85% say that the Intro to programming subject is very important for their course.
3) Related to the use of a methodology: 42% of the students assure they noticed that the teacher used some teaching methodology or practice.
4) Leadership: 73% of the students believe the professor of the subject possessed leadership in relation to the class.
5) Project: 65% of the students feel comfortable working in a research project under the professor with whom they took the Intro to Programming subject.
6) Language: 59% of the students believe that the programming language used in the subjected facilitated their learning process and understanding of the subject.

It is worth remembering that this is an initial evaluation/monitoring, which we intend to continue over the next ten years and establish if there was a change in the current scenario. We will also check if the change in culture and teaching methodology brings results, providing positive change in the subjects evaluation.

As future work, we aim to conduct further studies on failing and passing students, in different contexts and different domains aiming at:

1) Analyze the teaching corpus education for the work with the APC, CB and ICC subjects: seeking to identify the professor`s profile, good practices, and proposed activities for the teachers who teach these subjects.
2) Analyze the kind of education that the student had through middle and high school (such as the development of computational thinking, for instance) and the reflexes of this education on the students' journey.

The objective is to amplify this research and subsidize the elaboration of education propositions (for professors and students), methodologies, resumes, supporting material, infrastructure management, etc.

Acknowledgements

The authors of this paper would like to thank all the students of the University of Brasília (UnB) who answered our survey and sent us reports and improvement proposals for the research.

References

Al-Jishi, E., Khalek, N.A., Hamdy, H.M. *et al.* (2009). Students' perceptions of the efectiveness of a professional skills program in preparation for clerkship training. *Education for Health*, 22(2), 57.

Bennedsen, J., Caspersen, M.E. (2007). Failure rates in introductory programming. *ACM SIGCSE Bulletin*, 39(2), 32–36.

Bosse, Y., Gerosa, M.A. (2017). Why is programming so difficult to learn?: Patterns of Difficulties Related to Programming Learning Mid-Stage. *ACM SIGSOFT Software Engineering Notes*, 41(6), 1–6.

Cohen, P. A. (1981). Student ratings of instruction and student achievement: A meta-analysis of multisection validity studies. *Review of Educational Research*, 51(3), 281–309.

Edgcomb, A., Vahid, F., Lysecky, R., Lysecky, S. (2017). Getting students to earnestly do reading, studying, and homework in an introductory programming class. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, (), 171–176.

Ehlert, A., Schulte, C. (2010). Comparison of OOP first and OOP later: first results regarding the role of comfort level. *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*, (), 108–112.

Galbraith, C.S., Merrill, G.B. (2012). Faculty research productivity and standardized student learning outcomes in a university teaching environment: A Bayesian analysis of relationships. *Studies in Higher Education*, 37(4), 469–480.

Garrido, S., Oliveira, K.M., Rezende, F.A., Funghetto, S.S., Griboski, C.M. (2017). A expansão da educação superior no Brasil, a indução da qualidade a partir do SINAES e as novas perspectivas para a educação a distância. *Cadernos de Pesquisa: Pensamento Educacional*, 10(25), 19–35.

Hoffbeck, J.P., Dillon, H.E., Albright, R.J., Lu, W., Doughty, T.A. (2016). Teaching programming in the context of solving engineering problems. *Frontiers in Education Conference (FIE), 2016 IEEE*, (), 1–7.

Jenkins, T. (2002). On the difficulty of learning to program. *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences*, 4, 53–58.

Kahneman, D., Fredrickson, B.L., Schreiber, C.A., Redelmeier, D.A. (1993). When more pain is preferred to less: Adding a better end. *Psychological Science*, 4(6), 401–405.

Lahtinen, E., Ala-Mutka, K., Järvinen, H. (2005). A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin*, 37(3), 14–18.

Martins, S.W., Mendes, A.J., Figueiredo, A.D. (2010). Comunidades de Investigação em Programação: Uma Estratégia de Apoio ao Aprendizado Inicial de Programação. *IEEE-RITA*, 5(1), 39–46.

Moreno, L., Gonzalez, C., Castilla, I., Gonzalez, E., Sigut, J. (2007). Applying a constructivist and collaborative methodological approach in engineering education. *Computers & Education*, 49(3), 891–915.

Murphy, E., Crick, T., Davenport, J.H. (2017). An Analysis of Introductory Programming Courses at UK Universities. *Journal Art, Science, and Engineering of Programming – AOSA*, 1(2), article 18.

Pitterson, N. P., Brown, S., Villanueva, K. A., Sitomer, A. (2016). Investigating current approaches to assessing teaching evaluation in engineering departments. In: *Frontiers in Education Conference (FIE)*, *2016 IEEE*. 1–7.

Watson, C., Li, F.W. (2014). Failure rates in introductory programming revisited. In: *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*. 39–44.

Wiedenbeck, S. (2005). Factors affecting the success of non-majors in learning to program. In: *Proceedings of the First International Workshop on Computing Education Research*. 13–24.

Wilson, B.C., Shrock, S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors. *ACM SIGCSE Bulletin*, 33(1), 184–188.

E.D. Canedo holds a PhD degree in Electrical Engineering from University of Brasília (UnB), Brazil in 2012. Master by Federal University of Campina Grande, (UFCG) in Software Systems (2002). I'm graduated in Systems Analysis by University Salgado de Oliveira; Goiás (1999). I'm Professor of the Computer Science Course at University of Brasília – (UnB) since 2010. Her research interests include Informatics on Education, Software Engineering, Cloud Computing and Software Systems.

G.A. Santos is a full time Professor at University of Brasilia (UnB), Brazil, since 2010. He was a Professor at the Catholic University of Brasilia during 2003–2010. He obtained his master's in computer science in 2001 and currently is a PhD student at Electrical Engineering (UnB). His research interests include Informatics on Education, Collaborative Learning, and Educational Robotics.

L.L. Leite holds a PhD in Computer Science and has interest in the following research areas: Human-Computer Interaction and Computer Science in Education. She develops projects related to the use of technologies to support teaching and learning, aiming to decrease the dropout rates in Computer Science courses. Her projects includes topics such as Computational Thinking, Teacher Training, and Algorithms and Programming Education.