Performance Analysis of Computer Science Students in Programming Learning

Air Rabelo[1], **Luiz Claudio Gomes Maia**[1], **Fernando Silva Parreiras**[1]

[1]Department of Computer Science - FUMEC University
Av. Afonso Pena 3880 - 30130-009 - Belo Horizonte - MG - Brazil - +55 31 3269-5230

air@fumec.br, luiz.maia@fumec.br, fernando.parreiras@fumec.br

Abstract. *The difficulties faced by lecturers and students in order to teach and learn programming on computer science courses have been a research topic over the years. The hardship to understand the abstract and logic concepts and consequent demotivation has been resulting in high rates of novices' failure and class abandonment. This study adopted statistical concepts to analyze students' final grades in programming subjects and compare their performance. Data were gathered from a computer science course at a Brazilian University. The period analyzed was from 2010 to 2015 including six programming subjects from the first and second academic year. The results pointed a significant number of student failure (43%) and abandonment (25%). It was also discovered that even with different teachers, semesters and programming subjects, the students' performance mean were nearly equal. The discoveries of this work contributed to point the hardship faced by students and teachers to learn and teach programming.*

1. Introduction

The learning programming process is a challenge activity to novice programmers [Lahtinen et al. 2005]. The students from programming courses face hardship to understand and apply the abstract and logic concepts required [Medina et al. 2013]. The difficulties refers to syntax and semantics structures relating to programming languages [Xinogalos et al. 2015]. Piteira and Costa conduct a study and identified the greater difficulties perceived by students: designing a program to solve a task, understanding programming structures and learning the programming language syntax [Piteira and Costa 2013]. The learning process suggests use of practices to enable the students to deal with situations and problems in which an algorithm could be a solution [Medina et al. 2013]. Students identify the practical sessions as one of the most useful learning methods [Piteira and Costa 2013].

The issues faced by novice programming students, who are not able to cope with the natural difficulties associated, commonly leads many to demotivation [Gomes and Mendes 2014]. Some individual factors contribute to student engagement such as achievement, motivation and goals [Eltegani and Butgereit 2015]. There are frustration when the teaching does not meet students' expectations, which reduce engagement and affects the success or failure rates in programming [Eltegani and Butgereit 2015]. A research conduct by Eltegani and Butgereit found the following attributes required in programming learning: a)time to study; b)receiving feedback and support; c)intensive practice; d)reduce fear that is defined as lack of interest in programming; e)promoting self

confidence and problem solving skills; f)availability of devices and resources to practice outside class; g)feeling of pressure/no pressure; g)language used in teaching should be easy [Eltegani and Butgereit 2015].

The objective of this research is to perform a statistical analysis regarding the performance of computer science students in programming learning. The data were gathered from a Brazilian University, including first and second academid years' grades on programming subjects from 2010 to 2015.

The research questions are: a)What was the general performance of the students in the analyzed period? b)What was the failure and success rates? c)Was there a performance variation among students from first and second year, or from distinct teachers?

2. Methodology

This paper adopted statistical concepts to analyze students' grades of introductory programming subjects in a computer science course. The main object was to compare rates of failure and success. The data were gathered from a computer science course at a Brazilian University. The period analyzed was from 2010 to 2015 including six different programming subjects applied on the first and second academic year. In this Brazilian University the terms of programming subject takes six months and they are called as semester. There are two terms per year, one in the first semester ane other in the second semester. The data were organized and grouped by semester, such as first semester of 2010 and second semester of 2010.

The database had 4396 rows covering 8 semesters and six different programming topics. The data were disposal in the following columns: year and semester, names of subject, teacher and student, first test grade, second test grade, third test grade and final grade. The data were placed in a CSV (Coma Separated Value) file. The R programming language and software environment for statistical computing and graphics, and the R Studio IDE (Integrated Development Environment) was adopted to data manipulation and graphics generation. The CSV file was imported to R Studio and some R programs were developed to reach the results presented in this paper.

In the first graph was presented a histogram with final grades of all students and terms from the database. In second graph, full length database were used to present the following items: a)a global mean of students grades, b)a success mean rate, c)a failure mean rate, d)a abandon mean rate. From the third to fifth analysis the same items were demonstrated, but at that time the objective was to compare a performance analysis between different teachers, terms and topics. In the following analysis, a data sample imported to R Studio was used for the tests ANOVA (Analysis of Variance), Shapiro-Wilk, Levene and Tuckey. The object of those tests was to verify if there was any relevant performance difference between students from distinct semester and teacher.

3. Results of the Analysis

Including all data in the database, 4396 student grades in the first and second years on computer science programming subjects from 2010 to 2015, a histogram was presented in the figure 1 considering only the final grades. The grades go from 0 to 100, and students have to reach 60 at least to success. The highest bar displayed in the figure 1 represents

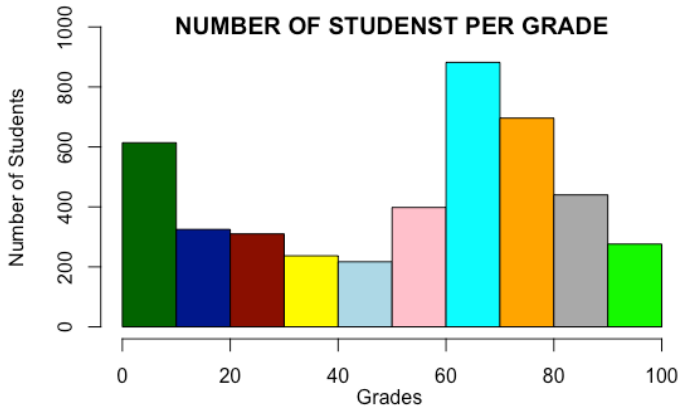

Figure 1. Histogram of Grades

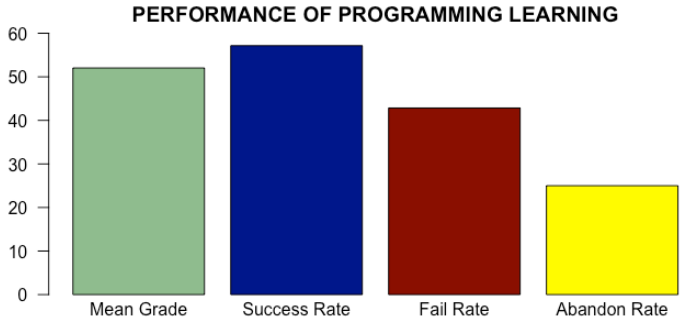

Figure 2. Students performance in programming learning

grades between 60 and 70, the next highest bar 70 to 80, they both were success grades. There were a significant number of students' grades from 0 to 59, which represent failure. This figure also revealed in the first left bar, grades from 0 to 10. There were around 600 students (in a total of 4396) in this situation of very poor outcome, which represent 13.6%.

In the figure 2 was presented the mean grade, rates of success, failure and abandoning. The expressive number of students with grades below 60% shown in figure 1 reduced the mean grade to 52%. The success rate was 57% and 43% of failure. Despite the success had been higher than fail, they were very near to each other. The abandoning rate was 25%, which means from those 43% of failure, 25% were students who had abandoned the programming subject before the end of semester. In this case those students quitted the class because they concluded themselves would not be possible to reach the minimal grade to success. The other 18% (43% - 25% = 18%) remained in the class until finish but failed at the end.

To verify if there were any difference on students performance considering the teachers involved in those programming subjects, the figure 3 exposed the mean grade, rates of success, failure and abandon, separated for each of the ten professors who taught programming classes in the analyzed period. The students of professors number 2, 5 and 9 had a failure rate higher than success. In the other hand, the students from the other seven professors' classes had just the opposite performance. The worst case scenario was presented in the professor 9 class, which the students had 32% of success and 68% of fail,

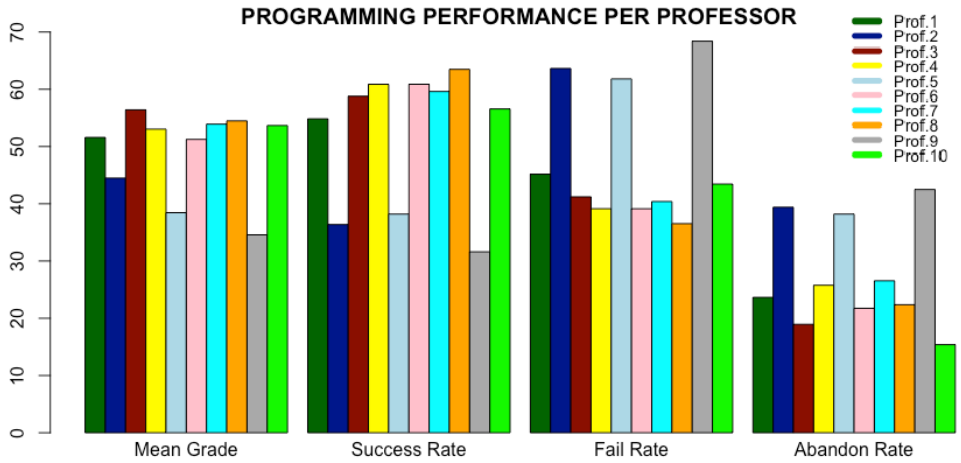

Figure 3. Students performance in programming learning per professor

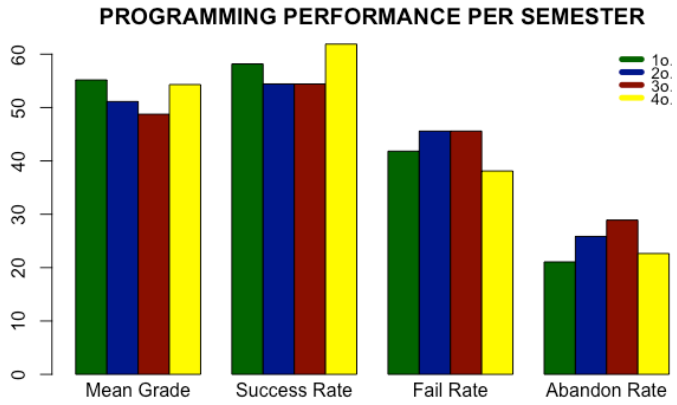

Figure 4. Students performance in programming learning per semester

42% of abandoning. The best case scenario was in the professor 8 class, which had 63% of success and 38% of fail, 22% of abandoning. In those cases the outcomes were almost the opposite each other. The reasons for that could be adoption of different assessment methods or criteria to assign grades. Despite those professors number 2, 5 and 9, in which students had more fail than success, in the others seven, which means 70% of professors, the students had similar performance.

A similar analysis of figure 3 was presented in figure 4. In this case the mean grade, rates of success, failure and abandon were presented per semester. The objective was to verify if students had different performance among the fourth semesters in which included the first and second years of the computer science course. Therefore, the figure 4 revealed a similar performance in all four semesters. Consequently, there was no indication that novice students from first semester had worst performance than others with some programming experience such as those from fourth semester.

In the figure 5 the mean grade, rates of success, failure and abandon were presented per subject. The subject Prog.I was from first semester, Prog.II was from second semester, Prog.III and Prog.V were from third semester, Prog.IV and Prog.VI were from fourth semester. The objective was to verify if students had different performance among those six programing subjects structured on the first two academic years of the computer

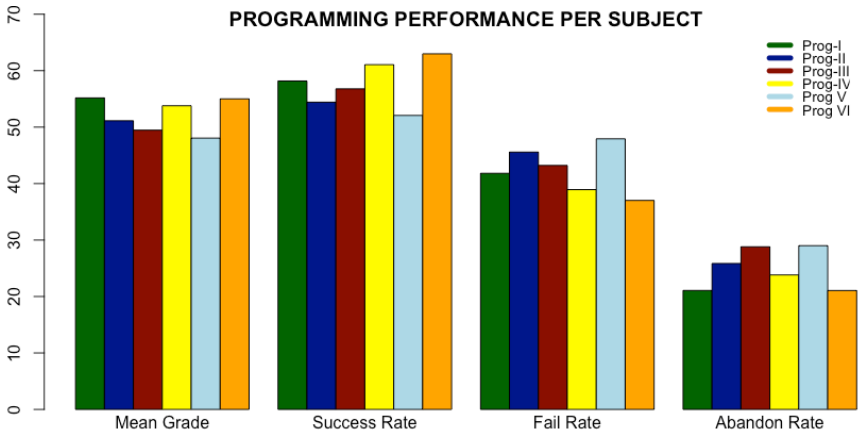

Figure 5. Students performance in programming learning per course subject

science course. In the same way of figure 4, the figure 5 shows a very similar performance of students between the six programming subjects analyzed. Despite the initial programming subjects such as Prog.I and Prog.II, in which programming concepts had less complexity than subjects Prog.IV and Prog.VI from fourth semester, students' performance was very similar among them. This indicated that as long as students gain more programming experience and apprenticeship, as time goes by, the complexity of the subjects also grew up, keeping the performance level similar between the six programming subjects analyzed.

To certify if the students performance were indeed similar considering different semesters and subjects, as it was presented in figure 4 and figure 5, a sample of database was isolated to perform an ANOVA test (Analysis of Variance). The ANOVA was used to compare means across groups of semesters and subjects. The ANOVA test was used to determine whether the data provided strong evidence of difference between the means. Otherwise, the null hypothesis is considered, indicating that all means are equal [Diez et al. 2012]. In the first sample was performed an ANOVA two-way. When an ANOVA is performed between groups using two factors it is called ANOVA two-way [Diez et al. 2012]. In this case, semester was one factor and subject the other. This sample gathered the six programming subjects in three sequential semesters: 2014 second, 2015 first and 2015 second. The figure 6 presented the box plot graph of the students' final grades mean on the six programming subjects across the three groups of semesters. The graph exhibits different mean grades among each group, 2015/1o was the lower one whilst 2014/2o and 2015/2o were similar.

The figure 7 presented the box plot of students final grade on the tree semesters across the six groups of programming subjects. The graph also exhibited different mean grades among each group, subject Prog I was the lower one and Prog VI the higher within more than ten percent points of difference. The ANOVA summary showed whether that differences were statistically significant.

The figure 8 exhibits the ANOVA summary for testing whether the mean of student final grade differs across semesters and programming subjects. The summary last right column showed the p-value. According to [Diez et al. 2012] whether the p-value is larger than 0.05, denotes that the evidence is not strong enough to reject the null hypoth-

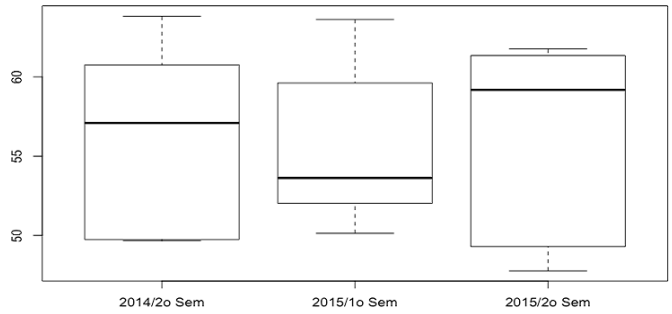

Figure 6. Side-by-side box plot of the students final grades on the six programming subjects across the three groups of semesters

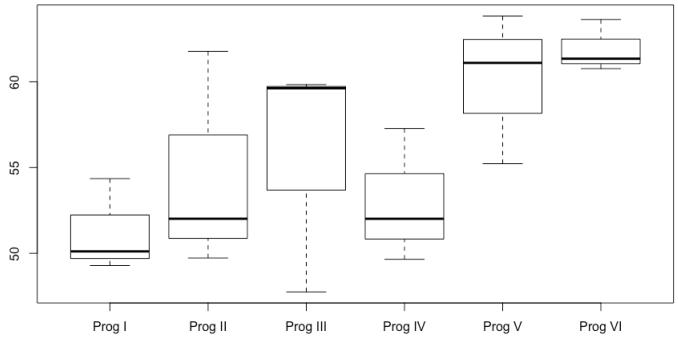

Figure 7. Side-by-side box plot of students final grade on the tree semesters across the six groups of programming subjects

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
tabela$Semester	2	3.66	1.83	0.07	0.933
tabela$Subject	5	256.13	51.23	1.95	0.172
Residuals	10	262.63	26.26		

Figure 8. The ANOVA summary for testing whether mean of student final grade differs across semesters and programming subjects

esis. The semester p-value was 0.933, which indicates a probability of 93.3% for null hypothesis, and subject p-value was 0.172, then the probability was 17.2%. Therefore, can be concluded that the data do not provide strong evidence indicating the mean of student final grade varied by semester or programming subject and the difference presented were due to chance.

As the first sample indicated an equivalent performance among students, in a second one was performed another ANOVA two-way also with semester and programming subject factors. This second sample gathered four subjects from first semester of 2012 to second of 2015. The figure 9 presented the box plot graph of the students' final grades mean on the four programming subjects across the eight groups of semesters. The graph also exhibited different mean grades among each group, 2013/1o was the lower one whilst 2014/1o the highest.

The figure 10 presented the box plot of students' final grade on the eight semesters across the four groups of programming subjects. The subject Prog II was the lower one and Prog VI the higher.

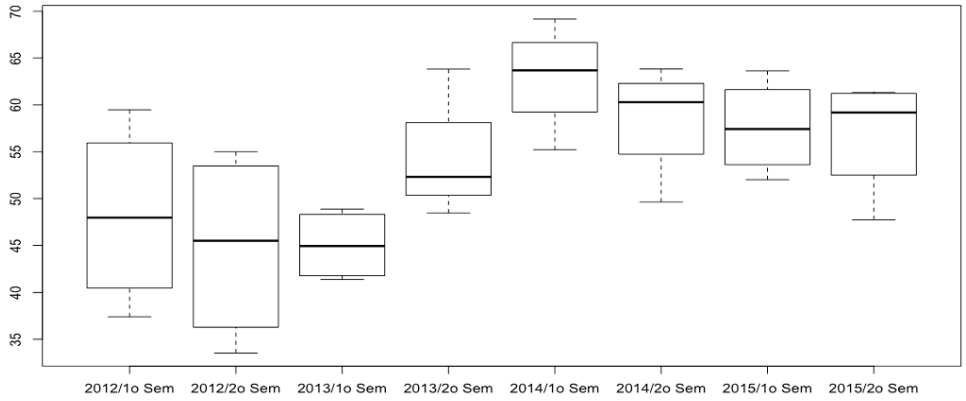

Figure 9. Side-by-side box plot of the students' final grades on the four programming subjects across the eight groups of semesters

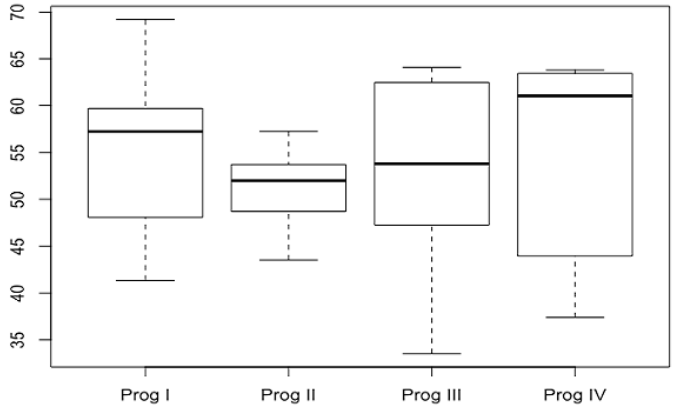

Figure 10. Side-by-side box plot of students' final grade on the tree semesters across the six groups of programming subjects

```
                Df Sum Sq Mean Sq F value Pr(>F)
tabela$Subject   3   76.3   25.42   0.480 0.6996
tabela$Semester  7 1267.8  181.11   3.421 0.0134 *
Residuals       21 1111.9   52.95
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 11. The ANOVA summary for testing whether mean of student final grade differs across semesters and programming subjects

The ANOVA summary exhibited in figure 11 showed whether that differences were statistically significant. The p-value for subject was 0.6996, which indicated a probability of 69.96% for null hypothesis. The semester p-value was 0.0134, so less than 0.05, then the null hypothesis has to be discarded and there was strong evidence that the different means in each of the eight semesters was not simply due to chance. In this case, in at least two semesters had been occurred different performance between students.

According to [Diez et al. 2012], there are three conditions to be checked for an ANOVA analysis: all observations must be independent, the data must be nearly normal, and the variance within each group must be homogeneous. To verify normality, a Shapiro-Wilk test was performed. The normality is reached when the p-value is higher than 0.05

```
        Shapiro-Wilk normality test

data:  resid(dados.anova)
W = 0.96787, p-value = 0.4427
```

Figure 12. The Shapiro-Wilk summary for testing data normality

```
> leveneTest(tabela$Performance ~ tabela$Semester, data=tabela)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  7  1.1467 0.3684
      24
```

Figure 13. The Levene summary for testing homogeneity of variance in semester

```
> leveneTest(tabela$Performance ~ tabela$Subject, data=tabela)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  3  1.0768 0.3749
      28
```

Figure 14. The Levene summary for testing homogeneity of variance in subject

[Shaphiro and Wilk 1965], as presented in figure 12 p-value was 0.4427, so the data were normal.

To verify whether the variance within each group were homogeneous, a Levene test was performed. The homogeneity is reached when the p-value is higher than 0.05 [Brown and Forsythe 1974]. As presented in figure 13 the semester p-value was 0.3684, and figure 14 presented the subject p-value 0.3749, so the variance of semester and subject groups were both homogeneous.

Having confirmed the normality of data and homogeneity of variance, to discover in which pair of semesters had a statistical significant difference, a post-hoc Tukey test were performed [Tukey 1977]. The Tukey test performed all possible pairwise comparisons within both factors semester and programming subject . The figure 15 presented the Tukey test summary with six pairwise tests of subjects and twenty eight of semesters. All the comparison between each programming subjects displayed the p-value higher than 0.05, pointing that this differences were statistically insignificant. On the other hand, from the twenty eight pairwise semester comparison, two had presented p-value lower than 0.05: 2014/1o - 2012/2o and 2014/1o - 2013/1o. It was a statistical significant evidence that the students from semester 2014/1o had a significant better performance in programming subjects comparing to 2012/2o and 2013/1o.

4. Conclusions and future work

This research pointed a significant number of student's failure (43%), which include an also expressive rate of subject abandonment (25%). The first ANOVA test performed analyzed means of student's final grades from three semesters and six programming subjects. The summary showed no statistical significant difference between those student's performance. Then, a second ANOVA and Tukey tests were performed including eight semesters and four programming subjects. It was pointed a difference of student's performance in two of twenty eight pairwise comparisons between semesters, which represents 7.14%. So, 92.86% of the students were considered to have had a nearly equal perfor-

Tukey multiple comparisons of means 95% family-wise confidence level

Subject	diff	lwr	upr	p adj
PROG II -PROG I	-3.8789871	-14.019935	6.261960	0.7132303
PROG III -PROG I	-1.9883564	-12.129304	8.152591	0.9464616
PROG IV -PROG I	-0.3201528	-10.461100	9.820795	0.9997475
PROG III -PROG II	1.8906307	-8.250317	12.031578	0.9534272
PROG IV -PROG II	3.5588344	-6.582113	13.699782	0.7631697
PROG IV -PROG III	1.6682037	-8.472744	11.809151	0.9672029

Semester	diff	lwr	upr	p adj
2012/2o Sem-2012/1o Sem	-3.3095244	-20.5673949	13.94835	0.9976730
2013/1o Sem-2012/1o Sem	-3.1683387	-20.4262093	14.08953	0.9982301
2013/2o Sem-2012/1o Sem	6.0254251	-11.2324454	23.28330	0.9315474
2014/1o Sem-2012/1o Sem	14.7376129	-2.5202576	31.99548	0.1313938
2014/2o Sem-2012/1o Sem	10.3166160	-6.9412545	27.57449	0.5016241
2015/1o Sem-2012/1o Sem	9.4178425	-7.8400281	26.67571	0.6081465
2015/2o Sem-2012/1o Sem	8.6631753	-8.5946952	25.92105	0.6969499
2013/1o Sem-2012/2o Sem	0.1411856	-17.1166849	17.39906	1.0000000
2013/2o Sem-2012/2o Sem	9.3349495	-7.9229210	26.59282	0.6180338
2014/1o Sem-2012/2o Sem	18.0471373	0.7892668	35.30501	0.0362584
2014/2o Sem-2012/2o Sem	13.6261404	-3.6317302	30.88401	0.1936296
2015/1o Sem-2012/2o Sem	12.7273668	-4.5305037	29.98524	0.2591516
2015/2o Sem-2012/2o Sem	11.9726997	-5.2851708	29.23057	0.3252506
2013/2o Sem-2013/1o Sem	9.1937639	-8.0641067	26.45163	0.6348321
2014/1o Sem-2013/1o Sem	17.9059517	0.6480811	35.16382	0.0384239
2014/2o Sem-2013/1o Sem	13.4849547	-3.7729158	30.74283	0.2029840
2015/1o Sem-2013/1o Sem	12.5861812	-4.6716893	29.84405	0.2707499
2015/2o Sem-2013/1o Sem	11.8315141	-5.4263565	29.08938	0.3387161
2014/1o Sem-2013/2o Sem	8.7121878	-8.5456827	25.97006	0.6913087
2014/2o Sem-2013/2o Sem	4.2911909	-12.9666797	21.54906	0.9888466
2015/1o Sem-2013/2o Sem	3.3924173	-13.8654532	20.65029	0.9972853
2015/2o Sem-2013/2o Sem	2.6377502	-14.6201203	19.89562	0.9994537
2014/2o Sem-2014/1o Sem	-4.4209969	-21.6788675	12.83687	0.9867605
2015/1o Sem-2014/1o Sem	-5.3197705	-22.5776410	11.93810	0.9634448
2015/2o Sem-2014/1o Sem	-6.0744376	-23.3323081	11.18343	0.9288251
2015/1o Sem-2014/2o Sem	-0.8987735	-18.1566441	16.35910	0.9999996
2015/2o Sem-2014/2o Sem	-1.6534407	-18.9113112	15.60443	0.9999758
2015/2o Sem-2015/1o Sem	-0.7546671	-18.0125377	16.50320	0.9999999

Figure 15. The Tukey test summary

mance. Therefore, regardless whether the programming subject is from first or second year of the course, or the semester analyzed, the means of student's final grades in programming learning had no statistical significant difference. It was analyzed whether there was any difference among mean grades considering different teachers, the majority of results also indicated a similar student performance.

The difficulty faced by novices on programming learning results in high rates of failure and abandonment. This has been addressed in several researches from different countries, such as [Tucker 1993], [Holland et al. 2009], [Stankov et al. 2015], [Guzdial and Guo 2014], [Robins et al. 2003], [Lahtinen et al. 2005], [Medina et al. 2013], [Xinogalos et al. 2015], [Miller et al. 1994]. Some of those researches also propose a methodology, computer tool or framework to optimize the effectiveness of teaching and learning programming. They also recommend a continuity of works in this area aiming to find better solutions to improve the student's performance in programming learning.

This work gathered data of students' final grades from a computer science course in a Brazilian university. Despite the data involved ten different teachers, the teaching methods were very similar and the assessment criteria were nearly equals. This fact favored the comparative analysis presented in this research and place reliability to discoveries reached, which corroborates the hardship of programming learning.

The data include eight semesters, six programming subjects from first to second year and ten distinct teachers. However, this study is limited by the period of the data analysed from 2010 to 2015 and by the one computer sicence course from one Brazilian university. Future works including undergraduate students from other courses and universities, covering a wide range of years, adopting different teaching methods, frameworks and other computing learning resources, would contribute to discover other results and complete the conclusions reached in this work.

References

Brown, M. B. and Forsythe, A. B. (1974). Robust tests for the equality of variances. *Journal of the American Statistical Association*, 69(346):364–367.

Diez, D. M., Barr, C. D., and Cetinkaya-Rundel, M. (2012). *OpenIntro statistics*. CreateSpace.

Eltegani, N. and Butgereit, L. (2015). Attributes of students engagement in fundamental programming learning. In *Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), 2015 International Conference on*, pages 101–106. IEEE.

Gomes, A. and Mendes, A. (2014). A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, pages 1–8. IEEE.

Guzdial, M. and Guo, P. (2014). The difficulty of teaching programming languages, and the benefits of hands-on learning. *Commun. ACM*, 57(7):10–11.

Holland, J., Mitrovic, A., and Martin, B. (2009). J-latte: a constraint-based tutor for java. In *17th International on Conference Computers in Education (ICCE 2009)*, pages 142–146.

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

Medina, C. F., Pérez, J. R. P., Álvarez Garc'ıa, V. M., and del Puerto Paule Ruiz (2013). Assistance in Computer Programming Learning Using Educational Data Mining and Learning Analytics. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '13, pages 237–242, New York, NY, USA. ACM.

Miller, P., Pane, J., Meter, G., and Vorthmann, S. (1994). Evolution of novice programming environments: The structure editors. In *of Carnegie Mellon University*, pages 140–158.

Piteira, M. and Costa, C. (2013). Learning computer programming: Study of difficulties in learning programming. In *Proceedings of the 2013 International Conference on Information Systems and Design of Communication*, ISDOC '13, pages 75–80, New York, NY, USA. ACM.

Robins, A., Rountree, J., and Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2):137–172.

Shaphiro, S. and Wilk, M. (1965). An analysis of variance test for normality. *Biometrika*, 52(3):591–611.

Stankov, E., Jovanov, M., Kostadinov, B., and Madevska Bogdanova, A. (2015). A new model for collaborative learning of programming using source code similarity detection. In *Global Engineering Education Conference (EDUCON), 2015 IEEE*, pages 709–715.

Tucker, S. A. (1993). Evaluation as feedback in instructional technology: The role of feedback in program evaluation. *Interactive instruction and feedback*, pages 105–132.

Tukey, J. W. (1977). Exploratory data analysis.

Xinogalos, S., Malliarakis, C., Tsompanoudi, D., and Satratzemi, M. (2015). Microworlds, games and collaboration: Three effective approaches to support novices in learning programming. In *Proceedings of the 7th Balkan Conference on Informatics Conference*, BCI '15, pages 39:1–39:8, New York, NY, USA. ACM.