International Journal of Enhanced Research in Management & Computer Applications, ISSN: 2319-7471
Vol. 4 Issue 3, March-2015, pp: (22-25), Impact Factor: 1.296, Available online at: www.erpublications.com

Experiences in teaching very large first year programming class

Sampson D. Asare[1], Zablon A. Mbero[2]
[1,2]Department of Computer Science, University of Botswana, Botswana

Abstract: In recent times, the number of students offering first year programming course in Java has skyrocketed significantly. This is as a result of the department of Computer Science having introduced three new programs; and all the first year students are required to take this first year course in Java programming. This has put pressure on resources in the way this course used to be offered, in terms of human resources, administrative and logistics requirements. In this paper, we share some of the experiences gained in teaching this course and also proffer suggested solutions to some of the challenges encountered.

1. INTRODUCTION

The department of Computer Science, University of Botswana, has embarked on expansion of its program, which started in the last 5 years. This expansion includes Bachelor of Science(Information Technology), Bachelor of Science(Computer Science); which was there originally. Others include Bachelor of Science (Computing with Finance), and Bachelor of Science (Information Systems). Originally, when we had only Computer Science stream, the intake into the first year was around 50 – 70 students. Once the expansion took effect this was quadrupled bringing in an entry figure of around 240 students (i.e. 60 per stream intake). However, this is not normally the case; students are admitted without regard to this upper limit. The consequence of this is that we do receive over 300 students cumulatively for first year pursuing one or the other of the above afore mentioned programmes, as was the case recently (August – December,2014), when we had roughly 341 students.

Previously this first year programming course had been taught using Python as a first programming language. When we took over, we changed the language to Java. Now it will interest one to know that this over 300 students all attend classes together in the same theatre. The theatre has no Public Address (PA) system or if it had one, it was not working or made available to the instructor. Naturally teaching a large class such as this one will entails some challenges and shortcomings. It will also require one to bring in past experiences, experimenting with new technologies, and strategies. In this paper, we share our experiences in teaching a very large first year programing class.

The next section discusses related work and experiences on others who have taught similar large classes and under similar or different circumstances. How did they triumphed and what we can learn from them. Section 3 clearly outlines the objectives of writing this paper. Section 4 discusses some of the challenges that we went through in the course of teaching this course, while section 5 is devoted to the way the course was administered and also implemented. Finally section 6 concludes the paper.

2. Related work

A number of researchers have shared their experiences in teaching very large classes (for the purposes of this paper, we accept very large class as any class over 100 students a in a single room, being taught by a single lecturer). Among these researchers and studies are the following: Isong E, Bassey [3] did a comprehensive study in which he elaborated on the best practices in teaching first year undergraduate students as a whole at Carnegie Melon University in the USA. Amongst the long list of best practices he dwelt on include but not limited to the following:

(i) Adjusting students expectations

(ii) Lecturers being approachable to students, removing all barriers so that students feel free to approach you as a lecturer with their concerns.

(iii) Getting students to come to class and on time , improving attendance and enhancing large lectures

(iv) Helping students manage and monitor their learning processes

(v) Engaging students in meaningful learning

(vi) Assuring quality in teaching

(vii) Addressing academic integrity issues

(viii) Challenging talented students

(ix) Helping students who are experiencing difficulties; recognizing and responding to students who may need assistance

In [4], Mario Cimino et. al. discussed how they have used Web-CAT system to enhance and improve the teaching of programming to large University classes. Web-CAT is an automated grading system for programming exercises licensed under the terms of GNU General Public License. Likewise [5] highlights the teaching of large classes across disciplines in Australia. In page 12 of the same document, they talked specifically about Computer Science large programming classes. The major problem faced by the teachers of large classes in this discipline is the disparity in students' background knowledge and computer literacy levels. Some of the remedies proffered towards this problem include the implementation of bridging courses, extra classes for less computer-literate students, telecenters, using cooperative learning techniques and problem centered project work in small groups. In [6], Jackson and Miller also shows us how they use both theory and practical in combination to teaching first year programming classes. They also highlighted on lecturing strategies and principles that they have used. Some of these include the CRP principle, which is Clarity, Realism, and Pragmatics which entails that each example that they give must be clear, and that each example must reflect a real problem (although not essential, but necessary) and finally the solution to each example should be solved pragmatically so that they appreciate how an engineer may solve his problem.

Another study was also done in the school of Mechanical Engineering at Chalmers University in Sweden, that focused on large classes. This covered first and second years[7]. Both Lecturers, "on the teaching", and students, "on the learning" participated, and one of the conclusions (pg. 19) that came out was that students expressed their willingness to adjust their learning approach so that it corresponds with the Teacher's approach to teaching large classes. In [8], a study was carried out that showed that students in introductory courses often have little or no prior knowledge about the topics or subject of study, and what they do know is also poorly organized incomplete or simply inaccurate ([3], page 2). They further argued that the lecturer teaching the course should be explicit about the expectations in order to deal with the presumptions the students may have. The lecturer should be approachable as most of the first year undergraduates are adjusting to many things that they are seeing for the first time; such as handling peer pressure, exploring new freedom, etc.

3. Objectives

Some of the main objectives in writing this paper, is to bring to light the challenges and experiences that we went through in teaching this very large class. By so doing, others can also learn how to handle large classes. The introduction of new streams and the combined first year program class for all streams seems to be a good idea, but the fact that one teacher handles all classes together does not appear to be a smart thing. To bring to light some of the challenges that we went through and possibly to find or proffer solutions to such challenges is also another goal/objective for writing this paper . Laboratory management as well as class handling also taught us some lessons. Clearly we have learned from our experience and can perform much better next time.

4. Challenges

A number of researches have been carried out on the correlation between large class sizes and its adverse effects on students by various authors. Our challenges are not overly different from theirs. For instance [1] mentions a number of such challenges as follows:

(a) That large class size increases faculty reliance on the lecture method of instruction

(b) That large classes reduce students' level of active involvement in the learning process.

(c) That large class size reduces the frequency and quality of instructor interactions with and also feedback from and to students.

(d) It was also found out that large class size reduce students depth of thinking inside the classrooms

(e) In addition, large class size limits the breadth and depth of course objectives, course assignments, and course-related learning outside the classroom

(f) It was also found that students' academic achievements; learning; and academic performance(their grade) are significantly lowered in courses with large class sizes.

(g) Also students tend to report less course satisfaction in large-sized classes, and hence tend to give lower overall ratings (in evaluating the course and the teacher as a whole) for courses having large students numbers.

In addition to the above challenges we faced a number of challenges that were unique to us. Some of the following were highlighted.

(a) Students were not all enrolled at the same time. Students kept enrolling into the course as far as 5 to 6 weeks after the class had begun. This meant that some of the students and this is a considerable number well into 100 students missed the very beginning concepts of programming, such as variables, identifiers, memory allocation, etc.

(b) Lack of Public Address (PA) system in the lecture hall meant that those far behind could not hear properly what the instructor was saying, even though PowerPoint slides were used, I had to occasionally write on the board to explain or get my point across.

(c) Controlling a class of over 300 students was not an easy task. Merely telling them to keep quiet sometimes did not work..

(d) Students attendance to both lectures and laboratory were at best less than satisfactory.

(e) Massive copying and cheating amongst students. This happened particularly in the laboratories, where they were given or assigned weekly exercises in addition to mandatory monthly assignments.

(f) Lecture period from 1 – 2 pm was not the best of times to schedule such a large class. Lunch hour , most of the students would be either hungry or tired and hence may not possibly concentrate on what is being taught

5. Course administration and implementation

The course, CSI141, Programming principles, is a first year programming course using Java as high level programming language. Previous lecturers have used Python to teach this course, but when we took over, we decided to change the language to Java. This was necessitated due to the fact that the second programming language course the students would be enrolling in, is conducted using Java as a programming language, hence we felt that it would be more appropriate to introduce the students to Java early on in their program.

It is a 3 credit course, which means we use two hours (lecture hours per week) and 2 hours laboratory hours per week per student. The two hour laboratory is considered as one hour lecture hour, hence the 3 credit hours. We had laboratory assistants who assisted us in the labs. Each laboratory consisted of either a 30 computer machines or 50 computer machines. Two laboratory assistants were assigned in charge of each room. Their main job is to assist the students during the 2 hour weekly period that they will be either practicing or working on their assignments.
We met the students between 1 to 2 pm on Mondays and Wednesdays. Laboratory periods were scheduled on Tuesdays from 7 am – 3pm, in 4 different rooms for 2 hours at a time running concurrently.

In addition to lectures and laboratory periods, we also used Moodle; a learning platform designed to provide educators, administrators and learners with a single robust, secure and integrated system to create personalized learning environments [2]. In Moodle, we arrange for students to submit their assignments, laboratory exercises, etc. We also put course materials such as course outline, course notes, handouts, exercises and tests so that those who miss lectures and labs can still access those materials.

In the administration of the course, we were mindful of some of the best practices that experienced lectures put into use and in fact had incorporated a number of them as outlined in [3]. These skills include but not limited to the following:

(a) Highlighting major points at the beginning of each lecture. As one professor puts it, “tell them what you are going to tell them, tell them and then tell them what you have told them’
(b) We also explicitly distinguished between generalization and examples, conclusions and evidence or trends and isolated events or incidents. This is important because first year students do not normally know how to differentiate between different kinds of information that are bombarded onto them. Specifically in

programming they are unable to differentiate the different concepts of terminology ; for instance the difference between a variable, identifier, memory space or allocation and values.

(c) Keep summarizing periodically during each class of what is being taught and also at the end, which is somehow similar to point (a) above.

(d) Despite the large class size, we also had to sometimes engage the students in an interactive session, where we stop, ask questions or ask someone to come up to the board to do some examples or exercises for the rest of class to see.

To improve class attendance, we also employed a number of strategies, such as

(a) Starting classes always on time, so that they get the impression of time promptness and consciousness.

(b) The course was structured in such a way that students were accountable for attendance we actually took attendance in the laboratory to give the impression that it will be used towards the end of the semester. Also lecture notes were not promptly updated on Moodle. We did this purposely so that students would come to class to gain understanding rather than thinking they can study on their own, which majority never did.

(c) We also stressed explicitly the importance of attending lectures and laboratories.

6. Conclusion - Lessons learned and ways to improve upon them

We have shared our experiences in teaching large classes, introduced the course Programming Principles, gone through related work by different authors, clearly outlined some of the objectives in writing this paper. We went on to discuss some of the challenges that we faced during the administration of this course. Under the course administration, we also talked about how to improve or remedy some of the challenges identified earlier on. We conclude that this sharing of our experience will contribute to the knowledge of teaching programming course and any other large class in a first year University classes that are large.

References

[1]. Joe Cuseo, The Emperical Case Against Large Class Size adverse effects on the Teaching, learning, and retention of first year students**.** URL: http://www.ulster.ac.uk/star/curriculum_development/cuseo_class_size.pdf

[2]. Moodle software. URL: https://moodle.org/about/

[3]. Isong E. Bassey, **Best Practices for Teaching First-Year Undergraduates**, strategies from experienced faculty, Eberly Centre for Teaching Excellence, Carnegie Mellon University, USA, Mellon , International Journal, Modern Education and Computer Science, 2014,, Vol(9), pp 15 – 21.

[4]. Mario G. C. A Cimino, Giuseppe Lettieri, Giovanni Stea, "Using Web-CAT to improve the teaching of programming to large University classes", DIDAMATICA 2013

[5]. Large classes across disciplines, Teaching and Educational Development Institute, Teaching Large classes , The University of Queensland Australia, 2001 URL: http://www.cadad.edu.au/largeclasses/pdfs/LitReview_5_Across.pdf

[6]. Daniel Jackson and Rob Miller, "A New Approach o Teaching Programming", 2009, URL:http://people.csail.mit.edu/dnj/articles/teaching-6005.pdf

[7]. Mikael Holmquist, Sven Andersson, Per-Ake Jansson et al, Large Class teaching/learning " A C-Selt Project At Chalmers University Of Technology, Sweden 2001/2002.URL: http://www.cdio.org/files/largeclass_tching.pdf

[8]. Susan Ambrose and Rea Freeman, Best Practices for Teaching First-year Undergraduates , Eberly Center for Teaching Excellence, August 1997. URL http://www.unomaha.edu/facconnect/BestPracticesforTeachingFirst.pdf