Students' Perception about Fundamental Programming Course Teaching and Learning

Nejood Eltegani
Sudan University of Science and Technology, Khartoum, Khartoum, Sudan
Email: nejood.phd@gmail.com

Abstract— Programming learning has unique characteristics as it is a subject that requires skill and higher order thinking. Students come to class with a perception about the subject mostly obtained from their seniors including fear or perceived difficulty. Senior students have a perception about programming learning that was supported by their experience during the subject learning. Students' views (+ / -) about the course could affect their performance. A qualitative survey was conducted with 93 third year students to obtain their views about the students' point of views while learning programming and the recommendation for modifying the course. Obstacles identified by students could be tackled with the aid of technology enhanced learning (TEL) including tutoring system. This survey is done as a preliminary step in developing and incorporating technical solution to students' problems. The findings were: Mostly, students are satisfied with the amount of time and effort they dedicated to the subject. While some mentioned that they would practice coding more and perform some projects beyond the course level. Majority of the students pointed out that they got useful advice from seniors about the subject learning. Less feedback was discouraging to students. About their suggested modification about the way the course setup, their overall responses approved the course design. There were minor comments about the proportions of the theoretical to practical components and the suitable amount of assignments.

***Keywords**- Students' perception, Fear, TEL, course design .*

*****

I. INTRODUCTION

Fundamental Programming is a subject that is required by almost all science and engineering faculties at Sudan University (SUST), and it is a major subject for computer science students. Studying programming requires interest, self-efficacy and discipline from students. Students differ in the way they learn and respond to learning methodologies and materials presentation styles. Several positive students' attitudes can aid in the process of learning. Many students lack the positive attitude and perception for programming learning as they approach the subject with perceived difficulty and fear.

It is difficult for many students to use programming languages to write programs to solve problems. One of the reasons that cause learning difficulty is the lack of problem solving abilities that many students show. Solving problems is not easy to learn and novices usually don't know how to create algorithms. Training is required in order to help students obtain that skill. In the work by Gomes and Mendes, they proposed building a tool that helps students practice developing and testing algorithms; their tool is named SICAS (Interactive System for Construction of Algorithms and its Simulation) [1].

In a study by Rogerson and Scott, data collected from students describe that programming concepts build up fast and that can cause fear for students. For their study, the word "fear" is regarded as a descriptor for denoting a lack of interest in programming as a discipline, lack of confidence or hesitation regarding their ability to code or program. This fear affects other aspects related to their studies, such as self-confidence, time management, and problem solving skills. For those students, there is a critical need for intervention, and some suggestions have been made. For example, formal, one-on-one consultations with the lecturer at strategic intervals may help students to overcome their fears sooner and, as a result, increase their comfort and enjoyment levels. The aim is to allow the students to reach their full potential without fear [2].

Programming is a craft that often demands that learners engage in a significantly high level of individual practice and experimentation in order to acquire basic skills. However, practice behaviours can be undermined during the early stages of instruction. This practice when left unchecked; create cognitive-affective barriers that interact with learners' self-beliefs which will potentially reduce practice. Scott and Ghinea seek to ascertain how to design a learning environment that can address this issue. They proposed that analytical and adaptable approaches, which might include soft scaffolding, on-going detailed informative feedback and a focus on self-enhancement alongside skill development, can help overcome such barriers [3].

Students may lake interest in any of the activities needed to learn programming and hence face difficulties in learning. It is worth noticing that the nature of the programming subject enforces great level of collaboration between learners. Students can benefit from their colleagues in obtaining help and explanation. However, studies show that many students are passive. In [4], suggested that the cause of students being passive is due to the fear of being judged or not having something interesting to say.

The research in [5] investigated the collaboration in fundamental programming, and their study showed that collaboration in fundamental programming learning might not be suitable. In their work they identified the difference between collaborative and cooperative work, with collaboration indicating no division of work among the team members. It is useful to have the work divided among the team members (individual practice) to simplified tasks.

The paper in [6] pointed out the difficulties IT students face when they are taught using traditional instructor-oriented methods. It advises the lecturers to adopt teaching philosophies that improve their students' learning satisfaction.

Giving choices in the learning materials and providing continuous support helps in programming learning. ICT offers countless solutions that can be utilized in education.

This survey aims to find students preferences and opinions about fundamental programming teaching and elements that they are hoping to get during learning.

II. METHODOLOGY

Answers to freeform questions were collected from 93 third year students who studied introduction to programming in their first year and they also studied additional programming courses in their second year.

To analyze the qualitative survey, the answers were studied carefully and the answers were assigned to broader groups of responses [7]. Word clouds were used to highlight the words that are more frequent in the responses.

The qualitative survey gave free form questions to the students to a) Activities students would have done if they had more time. b) The advice students had prior to studying the course. c) Describing their engagement level and the reasons that make some students disengaged and the suggested modifications on the subject teaching to make it more engaging, d) The students' study sources and their sources for getting help, and e) The advice that they have for future programming learners based on their experience.

a) What Activities would you do if you have had at extra time for study?

Students reach to conclusion about their performance in studying the various subjects. They are able to analyze the study plans they followed and the desired perceived level they reached. Students were asked about activities related to programming learning they would have done if they have had more time. Some of the students' feedbacks were:

"Nothing."

"I am saturated."

While others pointed out that they did well, still they could have done some projects or learned further concepts.

Some students pointed out that they could have done better and the following figure showed the most frequent words in their responses.

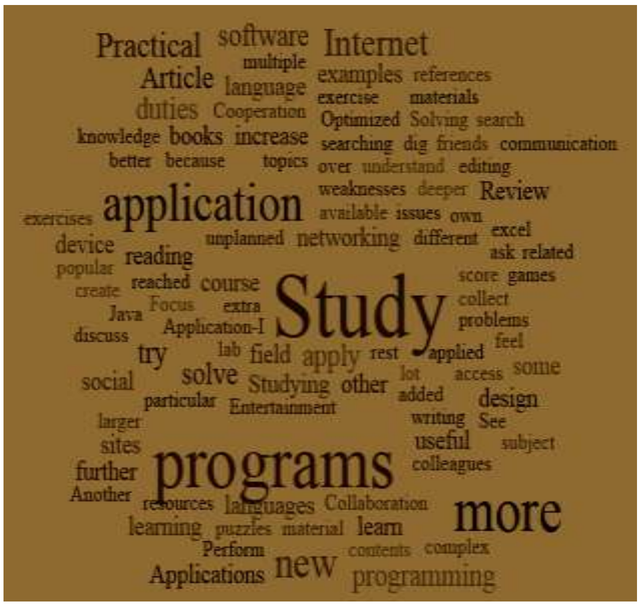

Figure 1: Activities students would have done in extra time

b) The advice you got before you start the course

Students tend to ask their seniors about the subjects they are about to start studying. The following figure shows the group of advice that were given to the group that is being interviewed. The majority of advice give positive direction such as apply coding, pay attention and follow closely. While there are fewer feedback that can cause fear for novices such as the subject is difficult and improve your English (main language for online materials and help).

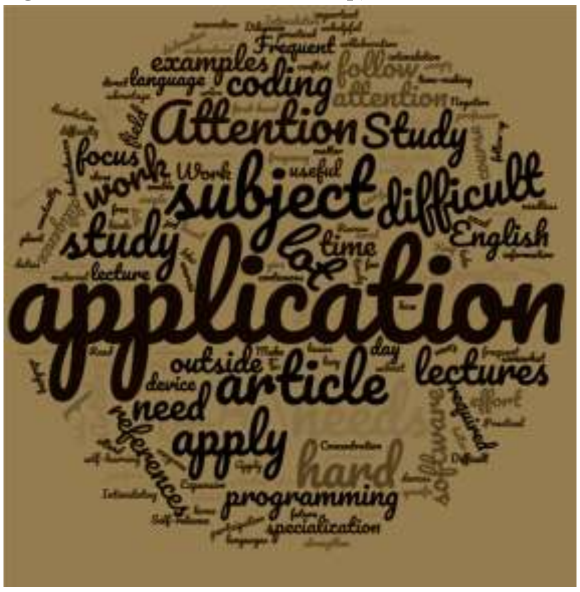

Figure 2: The advice novices received from seniors

c) Suggested Modification on Subject Teaching

Students were asked about how to make the subject more engaging. Their recommendations varied in some points, conflicting opinions were received from students who attended the same course about necessary changes in the course contents. The following opinions were among the top frequent responses.

- It is very essential to have plenty of practice in coding
- It would be good if there is a mean of running programs/examples during the lecture time.
- Students propose arranging competitions in coding and developing algorithms within university students and other universities competitions.
- Students believe that it is good to be able to view different ways of solving exercises. There should be sharing of solutions that can be seen by the students to take advantage of the presence of more than one way to resolve the correct software. Also there should be a vote or discussions about the best way of solving programs i.e. a winner program.
- Some students indicated that they were attracted at the beginning of the course, because they could understand the concepts and they could also meet deadlines in submitting the practical assignments. As the concepts accumulate, students feel that they need more time to solve assignments and they start to miss deadlines and lose interest to continue learning.
- Some of the students did not give opinion on making the material more attractive. This could be due to one of two reasons: firstly, the course is good the way it is (the students approve the current teaching method). Secondly, nothing can be done to make the learning programming engaging at all.
- Add more practical meaningful assignments or projects
- Don’t give complicated assignments so that students don’t feel unable and give up and hence copy solutions.
- Easy assignments so that students gain confidence in their level
- Break projects into modules with proper help when needed from tutors
- Students mentioned that they would like to see how a problem can be solved in different ways share and rank solutions or programming assignments.

The following table showed the categories of the responses about the suggested modification on the subject after analysing students’ feedback and grouping it into broader categories.

Table 1: Categories of students’ responses about the suggested modification

Suggestions group	**Count**
The simplified explanation and the presence of professors who are competent, sympathetic and available	15
Allocate more time for teaching programming	11
Focus on the practical aspect	10
Give several study sources	6
The subject is perfect the way it is no need for changes	6
Do programming contests , games and competitions	5
Support students to have their own devices	5
Assignments should have practical usefulness related to student environment	5
Reduce the time required for the material and the number of assignments	4
Execute examples in the lectures	4
Suitable syllabus	4
Increase the fun and challenge and avoid boredom	3
Study the subject in the lab	3
Give timely feedback for assignments and put extra effort on them	3
Do more tests	2
Tutors should allocate office hours in the lab with the students	2
The use of attractive presentations medium (make lectures interesting)	2
Use flow charts and algorithms for problems discussions	2
Use simple language to explain preferably Arabic language	2
provide recorded lectures	2
Reduce theory	1
Update syllabus and cope with market needs	1
Provide enough time in the computer laboratory	1
Give assignments with suitable difficulty so that students don’t feel frustrated and copy	1
Allocate some time to communicate with students online	1
Give extra optional assignments for the student to make up for missing assignments – no frustration	1
Reduce the load of other subjects in the semester so that students can focus on programming	1
Concentration of effort with weak students in the subject and give optional training for those for those who are willing to	1

As an alternative method of visualizing textual data, is to plot according to the word frequencies. In figure 1 the word list in the survey was plotted using word clouds. The key words can be seen in this cloud of words in which text size and colour darkness correspond to words frequencies. The tool used for plotting the figure is http://tagcrowd.com.

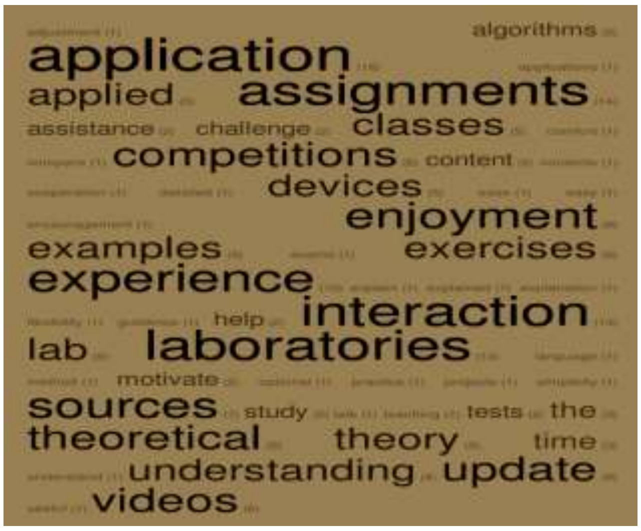

Figure 3: Suggested modification on fundamental programming subject teaching

In the previous figure the words were just shown without the desired action that is required by the students, namely the decrease and increase of words action. In the following figure, the previous data was plotted again showing the desired direction of actions + or -.

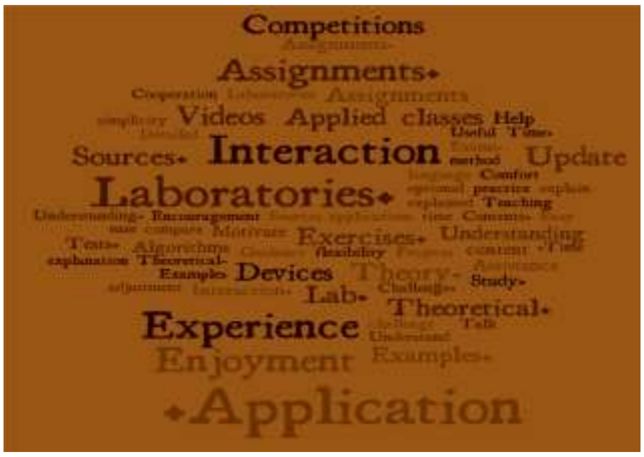

Figure 4: Suggested modification on subject teaching with desired action (+ or -)

The tool used for plotting the previous figure is WordItOut. In this figure only the size is correspond to the frequencies while the colour is assigned randomly for distinguishing words. The frequencies of some of the words were reduced as they were divided according to students' opinion. For example some students think that there should be less theory (theory-) while others prefer more of the theoretical lecture time (theory+). The same is applied for Assignments. In the other hand, most of the students prefer to have additional time practicing and coding not necessarily in assignments.

d) Sources for study and help

In responses to the question about students' studying resources and the sources of tutoring that helped students beside the teaching staff. Their responses showed that they benefited to great extent from colleagues and seniors. Students look for several resources about the same topics hoping to find the source that gives simple explanation to the topics.

Several activities can improve and accelerate programming learning e.g. practical assignments, tutorials , recorded audio and video revisiting materials can be like a second chance for students to understand the materials and make up for the wasted time and missed sessions. Interestingly, students might get bored doing same activity for a particular concept several times. So providing a range of resources can help students focus and revisit concept.

Table 1: Sources for help

Source of help	Count
Colleagues	42
Seniors	15
Friends	12
Online	12
Professor	5
Internet	4
References	4
Family	3
Videos	3

e) Students' advice for future students

Their advice for future programming course students were categorized and listed in the following table:

Table 2: Categories of students' Advice for novices

Advice from senior to novice programming students	Count
do plenty of coding and study lots of examples	28
Timely study and solving exercises	13
Attend lectures and labs and pay good attention to them	13
If this subject is mastered then the following years of study will be easier	5
Do plenty of search and trying to tackle difficulties	5
Depend on self study	4
Study references	4
Access to diverse sources	4
Don't fear from the subject difficulty	3
Use the lectures available online	2
Collaborate with colleagues	2
Application of useful projects	2
have positive qualities such as desire, challenge, ambition, research and diligence	2
Obtain your own device	1
Improve the English language level	1

The following figure 5 depicts the most frequent keywords in the advice they gave.

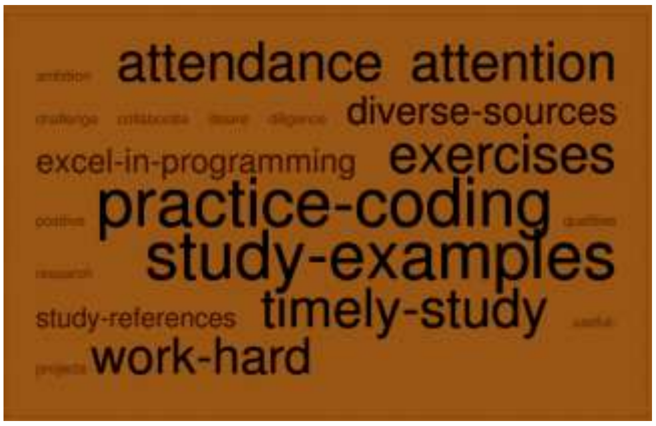

Figure 5: Figure of top words of students' Advice for novices (plotted using http://tagcrowd.com)

One of the questions for the students was about their utilization of the time they had, majority of the students admitted that they wasted time that they could allocate to studying. While few mentioned that they were saturated and that the effort they allocate to studying programming was satisfactory for them and they couldn't have done more about learning programming. This reflect that there is a room for enhancing programming learning but the choices of activities available for students were not engaging.

III. CONCLUSION

Students pointed out that it is useful to share coding solutions and the different ways for solving a problem. Students mention that it is good to have a feeling of competition in solving problems and ranking for best solution. It is very clear from the responses that this is a subject that needs close follow up of students during learning. Disengagement can result when students feel left behind and it can also happen for good students who feel that the subject is easy and boring. If timely doing assignments and close follow up is performed by the students this will have positive effect on the achieved level and subject time quality.

Tutoring system could be adopted and modified to aid in programming learning and to provide a platform for collaboration, sharing and scaffolding.

REFERENCES

[1] Gomes, A., & Mendes, A. J. (2007, June). An environment to improve programming education. In Proceedings of the 2007 international conference on Computer systems and technologies (p. 88). ACM.

[2] Rogerson, C., & Scott, E. (2010). The fear factor: How it affects students learning to program in a tertiary environment. Journal of Information Technology Education: Research, 9(1), 147-171.

[3] Scott, M. J., & Ghinea, G. (2013, April).Educating programmers: A reflection on barriers to deliberate practice.In Proc. 2nd HEA Conf. on Learning and Teaching in STEM Disciplines (p. 028P).

[4] Baccarani, C. (2014, December). Is the professor still useful in the age of the Internet?. In Liverpool (2014): 17th Toulon-Verona Conference" Excellence in Services".

[5] Imad, E.Z., Assessment of Collaborative Methods in Learning Principles of Programming., The International Journal of E-Learning and Educational Technologies in the Digital Media (IJEETDM) 1(4): 203-208The Society of Digital Information and Wireless Communications (SDIWC), October - 2015 (ISSN: 2410-0439)

[6] Chuang, Y.T., 2015. A teaching philosophy that utilizes instructional technologies to improve learning motivation. The International Journal of E-Learning and Educational Technologies in the Digital Media, 1(2), pp.121-132.

[7] Renner, M. and Taylor-Powell, E., 2003. Analyzing qualitative data. Programme Development & Evaluation, University of Wisconsin-Extension Cooperative Extension.

[8] WordItOut.com

[9] http://tagcrowd.com