Journal of Engineering Education Transformations ,
Volume 31 , No. 1, July 2017, ISSN 2349-2473, eISSN 2394-1707

A Practical Approach to Make Computer Laboratory Courses more Effective and Interesting through Student Mentoring

Uma Boregowda[1] , Sunitha P[2], A Geetha Kiran[3]
[1,2,3]Department of Computer Science and Engineering,
Malnad College of Engineering, Hassan, Karnataka, India
[1]umaboregowda@gmail.com
[2]suniachar2gmail.com
[3]geethaagk@gmail.com

Abstract: The prime job of an engineer is to manipulate energy, materials and information in order to create facilities and benefits for mankind. To perform this successfully, an engineer needs sound knowledge which is beyond mere theory and it also encompasses other life skills. The laboratory courses are an important part of professional and engineering undergraduate education; and they are the ideal place for active learning. Programming is an essential skill that must be mastered by anyone interested in studying computer science. Our experiences in teaching have proven that significant number of students find programming to be difficult and disheartening. This underlines the need for making the programming laboratory courses more effective, interesting and encouraging. In an attempt to achieve this, we have experimented students mentoring schemes during the laboratory classes. Student mentoring is just a form of support given to students, which has shown positive impact on their learning, thereby improving the quality of the engineering graduates. Few other methods are also proposed to make the laboratory course more professional and exciting.

Uma Boregowda
Department of Computer Science and Engineering,
Malnad College of Engineering, Hassan, Karnataka, India
umaboregowda@gmail.com

I. Introduction

Engineering is a practicing profession and is incomplete without laboratory practice. The overall goal of engineering education is to prepare students to practice engineering and in particular to deal with the nature of problems faced by the society. In a laboratory course, students learn in a real world environment, work as team members, discuss planning of experiments, share ideas about analysis and interpretation of data and/or results. Thus several factors that are learned during the laboratory work are – designing, creativity, data analysis, team work, effective communication and ethics.. Most engineering learning takes place in the laboratory and it demands the active use of knowledge and skill. For students, the objective of engineering laboratory is to "practice by doing" and it even helps them to gain better insight and understanding of theoretical concepts taught in theory courses. It also gives ideas for innovations.

Although much attention has been paid to curriculum and teaching methods, relatively less has been discussed and written about laboratory courses. In many cases, course objectives for laboratory course are not stated. The skills essential for programming are problem solving and analytical skills [1] and hence the major theme of a computer course must be to emphasize on these skills.

Programming is a process of problem solving and clear thinking in order to transfer step-by-step

procedure from a natural language to a program. This is not a very easy task to be carried out without proper training and practice.

Programming laboratory courses have not received deserved attention in recent past. The usual way to conduct the course is to set a list of programs to be carried out, optionally supported with manuals. It is practically very difficult for two faculty assigned to a laboratory course with around 35+ students to interact with each student, encourage and support them to develop their own program and to evaluate the developed program.

An effective feasible solution proposed and experimented is to use few selected senior students as mentors, who will provide additional support to the students. Mentors will test the program for all possible inputs, explain the program logic if required and provide additional information on programming and software tools. This process will benefit both students and mentors in several dimensions, hence the institute in large. The details of this scheme carried out in our institute along with observed outcome is elicited in this paper. Few other methods proposed to improve programming are also listed.

II. Background

Many Students believe that programming skill is complex and hence they do not put sincere efforts to understand it.

According to Robert[2] the main source of difficulty is not only in understanding the syntax or concepts of a programming language, but rather on how programming constructs can be combined to form a program to accomplish the given task. There is a need to look out for a learning or teaching strategy which supports significant number of students to acquire programming skill.

There exists a long tradition of using undergraduate students as Teaching Assistants(TA) [3] since 30 years mainly as a cost cutting measure. But Reges[4] was the first to present their experience. Stanford continued this program for several years and streamlined the entire structure and recorded the observed outcomes[3]. Their findings showed that this increased the rapport between Teaching Assistants and students, and that TAs often acted as role models for students. The University at Buffalo used undergraduate TAs specifically to enhance the level of teaching[5]. It is observed that several institutes use undergraduate TA for larger classes where it is very difficult to run the courses without them.

All the above mentioned papers describe the success of using undergraduate TAs always in the context of providing required additional help for the professor. But we have used undergraduate students as mentors for laboratory course, and not as TAs in our small college environment. And the goal is to provide additional help to students for better learning than reducing the workload of professors.

III. Students Mentoring Scheme

Mentoring is a two-way street where both participants give and take, learn and grow; and enhance their professional skills. The use of students as mentors is a feasible and an effective approach. There are two possible ways to implement the idea – peer mentors and senior students. The peer mentoring can be easily implemented in a non-formal way by asking few students who finish their assigned program quickly to assist those who are struggling to get the program correct. It would be difficult to get significant number of students completing the programs quickly to help others during each laboratory class. Also available time would not be sufficient to render good support to slow learners. Hence, this method does not guarantee required support to all needed students during all laboratory sessions.

But senior students can be used effectively for mentoring. The details of this scheme are outlined below.

A. Structure of Senior Students Mentoring Scheme

Few volunteered and handpicked senior students are chosen to work as mentors. Each mentor is assigned 5-6 students. Their work is to assist students during the last one hour of regular laboratory time slot.

B. Role of Student Mentors

A mentor should extend student support and guidance in the programming area. He/She should be a positive role model, knowledgeable and skilled. They should help students develop and enhance their programming skills and confidence. In addition, they should provide the appropriate level of supervision and assist with planned learning experiences. The

following tasks are to be undertaken by a mentor for the programming laboratory classes.

- If a student has completed the program, check the output for all possible cases. Meantime, explain the need of testing and how test cases are chosen. For students who have not completed program, provide enough support so as to make him complete the program on his own.
- Ask students to explain the algorithm used. If the student fails to explain, then tutor will make best efforts to explain and make him understand the algorithm.
- Give a related problem to be solved, which needs slight modification to the completed program.
- Mentors must teach methods to debug the program and make the students to actually perform debugging by explicitly introducing bugs if needed.
- Whenever time permits, talk on good programming practices they have learned, other applications/tools to be explored and share their experience of learning.
- Make attempts to convince slow learners that everyone can learn programming provided they put increased efforts. Motivate students to work hard to stretch a little more.

C. Role of Faculty

- Faculty will give problems to be solved in that class either before or during the class.
- The given problem shall be explained and if needed, the algorithm and additional instructions.
- Mentors work should be coordinated and monitored. Whenever necessary, extend support and give guidelines to mentors.

D. Role of Students

- Slow learners must utilize this opportunity to better understand the algorithm and programming techniques.
- Good learners must interact with mentors to know about other avenues to be explored and learned in programming and technology.

Although it might seem that student mentors are loaded with this additional task, in reality they are also significantly benefited. Each mentor will have to work for only 1 or 2 hours per week.

E. Advantages to Students Mentors

Following are the benefits to mentors:

- Process of teaching someone will certainly help one to better understand the concepts and enhancing their own programming skills.
- Communication skill will be improved
- Contributes to the mentor's own personal and professional growth
- Enjoy better relationship with junior students
- Faculty and students will be recognizing them with high privilege, at times by parents of students also
- Provides intrinsic satisfaction by helping an emerging professional develop to his/her potential
- Professional distinction as someone who can serve as an example and role model for others

Student mentors are also given an incentive to boost their interest and their confidence level in mentoring their juniors. In turn, this strategy helps juniors themselves to learn better and become mentors for next generation students.

IV. Few Other Methods To Improve Programming

Our experience with students and doing recent online courses, have helped us to experiment and propose few other tips to improve the learning in a laboratory course.

The problem statements given in laboratory class must be related to real life situations, so that it becomes interesting and challenging to students. Solving these kind of problems, will help students to understand which technique and algorithm to be used when encountered with real life problems. Faculty must avoid giving exactly the same set of problems every year.

Students must be made to choose appropriate names to variables used in the program, instead of always using random ones like – i, j, k, a, b etc. They must be made to realize that proper naming of variables would help one to understand, debug and maintain the code. It must be made mandatory to use appropriate comment statements in every program.

The output of each program must be checked for large set of data, simulating real life situations. Hence, program must be written to take the data from a file instead of few numbers from the console. This better prepares and boosts confidence of students to handle

real life problems in their career.

For fundamental programming course, most students struggle to write even simple program on their own. Most spend time only on writing few input and output statements and do not attempt to write the actual code to perform the task. This problem can be overcome by giving the skeleton program which has initial declarations of all variables, input and output statements. The student is now forced to work on writing the code to perform the actual task and he will be successful in many cases. It is difficult to check the program for all cases for everyone, hence the list of possible cases with input values and expected output values must be given to students. Now, they can check their program for all the given cases themselves with little or no support from faculty.

V. Results and Discussion

Regular constructive feedback on the performance of students helps the faculty to assess student fairly. Mentors also give regular feedback on aspects like programming knowledge , interpersonal skills, attitude, problem solving techniques. Continuous assessment is done by having regular 'snapshots' of students' practice and feedback is given frequently. In addition, Informal feedback is also gathered along with these formal sessions which is essential for student development.

The proposed approach of mentoring students by senior students was experimented for fourth semester students for the laboratory course Data Structure with Object Oriented Programming in Information Science department. The number of students in the class was 47 and 8 student mentors were chosen by the faculty.

In the beginning of the semester, student mentors were explained the responsibilities and importance of their role. They were given instructions and tips to come better prepared for each laboratory session.

Two ways to measure the outcome of any experiment are direct and indirect methods. The direct method to check the impact of the proposed scheme is carried out by comparing the examination result with previous year without mentoring. The graph in Figure 1 shows the result of the laboratory course "Object Oriented Programming for Data Structure" for four consecutive years. It is evident that mentoring scheme has improved the results marginally across all categories of students. This is an indicative of the positive effects of adopting student mentoring scheme in teaching practice.

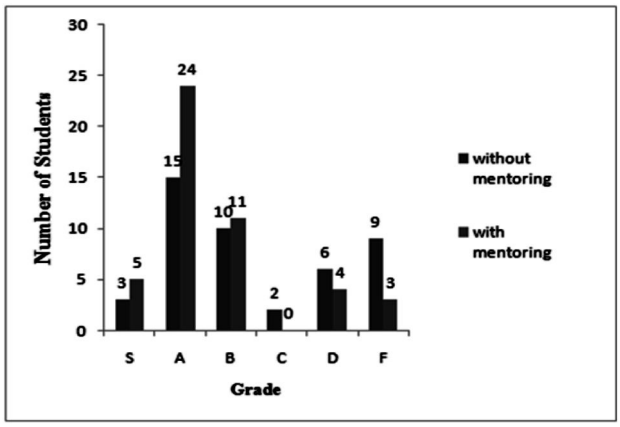

Figure 1 :Results of Proposed scheme and Examination

Further, to better analyze the effectiveness of the mentoring scheme, responses were gathered from various aspects like feedback from students, mentors and faculty about the technique employed and its impact on students.

Students expressed their happiness about the tutoring program they underwent. As an indirect method, responses from students about tutoring scheme were collected using Google forms.

The questionnaire included five Likerscale questions and one open question regarding the student's expectation of the tutoring program (based on 5 scale system: 1 being the least and 5 being the maximum). Further, it reflected their individual level of satisfaction and needs. The criteria for evaluation and the results are tabulated in the Table 1.

Table1: criteria for evaluation and the results

Sl. No	Criteria	Average (Standard Deviation)
1.	Influence of tutoring program on my success in the course	3.75(1.02)
2.	Helped in building a strong foundation of programming and algorithm design	3.82(1.4)
3.	Clarified my doubts and solved my problems	4.1(0.7)
4.	Enhanced my interest and confidence in programming	4.2(1.3)
5.	Meeting my expectations	3.6(1.8)

Tutoring program has also helped to achieve

several graduate attributes which otherwise would have been very difficult or impossible to achieve for laboratory course. The achieved graduate attributes because of the tutoring program are :

- Improved communication skills
- Professional and ethical responsibility
- Lifelong learning
- Contemporary issues

The faculty also gave a positive opinion that tutoring program improved programming ability of students.

Mentoring even benefits the mentors as much or more than it benefits the mentees. The responses were compiled based on the following points.

- Helped me to work effectively with other individuals
- Helped me to improve good communication skills
- Need of prior preparation of the laboratory programs
- Enhanced their programming skills
- Value of this mentoring scheme to you
- Usefulness of this mentoring strategy

The compiled responses clearly showed that this mentoring scheme has helped them to improve various aspects of their professional skills.

Feedback from students on mentors was also collected through Google forms focusing on the following points.

- Mentor's response and ability to clarify doubts
- Preparedness of the mentor in prior
- Need of prior preparation of the laboratory programs
- Helped me to improve my competency
- Mentor's teaching and communication skills

The compiled responses were overwhelming and satisfactory. It helped the faculty to choose able student mentors to make this mentoring scheme still more effective to the student mentees .

Mentors also gave their feedback on their group of mentees which helped the faculty in-charge to improve the programming ability of the students who are below average. The responses were more focused on mentees willingness to participate and learn, their satisfaction, behavior, attitude and keen to succeed. The chances to help their juniors experiencing many of the problems that the mentor's experienced at that stage of their careers was also evident.

For many, the experience was not only gratifying, but outcomes were more tangible. Consequently, the proposed mentoring technique has benefitted all the three components i.e., faculty, student mentors and student mentees.

V. Conclusion

Our experience of using student mentors for laboratory course was a fruitful one, the improved results and positive responses were its justifications. The process benefitted students, mentors and institute in large. Students get an opportunity to interact with more knowledgeable senior students, which they can utilize in several other aspects. We felt the need to train and guide mentors, so that they can be more effective and productive during the sessions. To attract better students for tutoring, some incentives must be given to them. We must work out these aspects and also give a more formal structure to the whole process.

References

[1] Linn M C and Clancy M J, (1992) The case for case studies of programming problems, Communications of ACM, 35(3), 121-132.

[2] Robert S. Rist, Teaching Eiffel as a first language, Journal of Object oriented Programming, 9, 30-41, 1996.

[3] Eric Roberts, John Lilly, and Bryan Rollins (1995) Using undergraduates as teaching assistants in introductory programming courses: an update on the stanford experience. In SIGCSE '95: pages 48–52, New York, NY, USA, ACM.

[4] Stuart Reges, John McGrory, and Jeff Smith (1988) The effective use of undergraduates to staff large introductory cs courses. SIGCSE Bull., 20(1):22–25,.

[5] Adrienne Decker, Phil Ventura, and Christopher Egert (2006) Through the looking glass: reflections