Enhancing Student Learning and Engagement in Freshman Course on Problem Solving using Computers

Jeyamala.C[1] & Abirami A.M[2]

[1]Department of Information Technology, Thiagarajar College of Engineering , Madurai, Tamil Nadu
[2]Department of Information Technology, Thiagarajar College of Engineering , Madurai, Tamil Nadu
[1]jeyamala@tce.edu
[2]abiramiam@tce.edu

Abstract: The paper presents a detailed impact analysis of incorporating appropriate active learning strategies for enhancing student learning and engagement in the first year introductory course on Problem Solving using Computers. The major challenge in Education system is that, most of the graduates learn by rote all the way from school to college. Most of the students do not have prior experience in programming and exhibit an aversion towards programming. Active learning techniques have been proved as a viable solution to eradicate rote learning. In order to promote higher order cognitive skills and student engagement, a systematic plan incorporating appropriate active learning techniques has been designed. The experimental study has been carried out in a first year student group comprising of 119 members from the Department of Information Technology at Thiagarajar College of Engineering, Madurai. Along with traditional assessment strategies, exclusive rubrics have been designed to measure the learning outcomes.This blend of instructor led and inquiry based teaching practices in the introductory course on Problem Solving using Computers has shown positive impact on student learning and engagement. Compared with the previous academic years, Students' performance in continuous assessment and End Semester Examinations has improved significantly. Positive feedback from students and increased count of participation in programming contests and online certifications demonstrate the improved effectiveness of the adopted teaching learning strategies in promoting self learning.

Keywords: Computer Programming, Problem Based Learning, Active Learning, Higher Order Thinking skills, Self learning

Jeyamala. C
Department of Information Technology,
Thiagrajar College of Engineering, Madurai.
jeyamala@tce.edu

1. Introduction

The freshman course on Problem solving using computers is intended to introduce about computational thinking and the methodology of programming with emphasis on modularity. The course on Problem solving using computers cannot be handled using the conventional method of lecturing alone (Gottfried, 1997). It has been pointed out that,

- Traditional 50 minutes of lecture
- Detailed programming examples on the board
- More emphasis on syntax ignoring program design
- More emphasis on program design ignoring syntax
- Individual assignments
- Unrelated in-class activities with the assignments

will not work effectively for handling a computer programming course. As many of the learners are new to computer programming, significant time must be spent on "learning by doing". Also, feedback by the instructor on every stage of learning is crucial to avoid misconceptions in learning. The limited meeting time for the instructor and students on a weekly basis is the major challenge for increasing student learning and engagement.

2. Related Works

Active learning has been strongly recommended by many of the education researchers to promote student engagement and learning (Bullard et al., 2008; Felder et al., 2009; Hake, 1998). Astrachan et al., 2002 have studied the effect of using collaborative learning in groups to increase participation and interest of the learners. Learners who were reluctant in class were able to discuss the solution freely in the groups. Female students made a remark that collaborative learning has made them comfortable in their classroom. Learners preferred group problem solving rather than lecturing. A novel practice of incorporating active learning to traditional in class labs led to an improvement in student outcomes (Briggs, 2005). Every lab exercise in the lab manual started with a brief description of concepts required for the experiment. It includes a narrative or sample code and make the students to get actively engaged by predicting the output, solving a problem or by completing a partial code. Learners extend the lab manual

by solving the set of exercises given at the end. The active learning model proposed by Hazzan et al., 2011 was adopted in the course on introductory programming. (Zimudsi, 2012). The implementation consists of four stages namely trigger, learning activity, discussion and summary. The roles played by the instructor and learners are elaborated.. A variant of Problem Based Learning namely Adapt, Design, Programming and Testing (ADPT) is experimented for computer Programming (Claudia & Marcia, 2014). Each stage of the previous model has a strong assumption that the previous stage is complete in all aspects. ADPT model promotes interaction among teams for better learning. Learners get familiarized with the software life cycle by developing an application from its conception to operation. ADPT is implemented in order to address CDIO standards. Active learning strategies like using visual examples, games, linage with known examples like cooking a sphaggeti, making a telephonic conversation etc., are incorporated in the introductory computer programming course and are found to have promising effects on student learning (Duffany, 2017). To reduce the high dropout rates and poor final grades, the separation between the lab and lectures of programming course has been removed. (Ebert 2017). Lectures were refined with more of practice using a web based platform. Use of such platforms enabled the deployment of activities related to the programming processes and concepts. The perception of students on the effectiveness of three instructional practices namely mini lecture, live coding and in-class coding has been analysed. (Adalbert et al, 2018). Experimental results show that students have a strong preference for mini-lecture and live coding than in-class coding.

It could be inferred from the literature that many variations of the active learning strategies are successful in enhancing student outcomes in the course on Computer Programming. The freshman course on computer programming aims at providing the necessary foundation skills for building a career in software engineering. However, recent report by the Economist magazine states that only 25% of the Indian graduates are employable. According to the study by Mettle, only 5% of the graduating engineers possess the analytical skills required for software engineering jobs and for product building. The reason for the poor skill development is because of the practice of rote learning and hence the learners are not able to meet the modern technological skills required in the work place. The second challenge is make the learners unwind from the practice of "rote learning" and imbibe "learning by doing" to improve their higher order thinking skills. The analysis of student intake in any Computer Science and Information Technology (CS&IT) programs reveal that, most of the learners have a passion for studies related to non-computer science and engineering. They are forced to select CS & IT programs because of parental and peer pressure. Hence a strong aversion towards computer programming from the students can be experienced especially in the first year of study. The count of failures in the courses on Engineering Mathematics and Problem Solving using Computers is significantly high in the last four academic years in the program under study.

The above mentioned challenges and limitations have motivated for a systematic plan including appropriate active learning strategies to enhance student learning and engagement. The instructional design should not only focus on improving programming skills, but should also promote the necessary skills and motivations for self learning. This research is an experimental study on incorporation of various active learning techniques with reference to the cognitive level of course outcomes. Effective assessment techniques have been incorporated to provide constructive feedback to the learners. An environment where the learners can create a repository of python codes for simple applications has been created.

3. Research Questions

The motivation for research is supported by the following research questions:

RQ1: What is the impact of active learning strategies on enhancement of learning outcomes, student engagement and student feedback?

RQ2: Does incorporation of active learning techniques provide the necessary foundation for promoting self learning skills?

4. Methods and Materials

A. Course Details

The introductory course on Problem Solving using Computers provides the necessary foundation for promoting problem solving and critical thinking skills. Upon completion of the course, the students would be able to master the principles of high-level programming and demonstrate significant experience in problem solving. The learning outcomes of the course, mapped in accordance with the Bloom's taxonomy in cognitive domain are presented in Table1.

Table 1- Course Outcomes mapped with Cognitive Levels of Blooms Taxonomy

COs	Course Outcomes	Bloom's Level
CO1	Comprehend the following terms in the context of problem solving by a computer: Problem specification, input-output analysis, algorithm, flowchart, pseudo-code, High level language, assembly language, machine language, and compilation and execution.	Apply
CO2	Write Python programs using appropriate programming concepts such as objects, data types, expression statements, branching and looping evaluation to solve a simple engineering problem.	Analyze
CO3	Select problem solving strategies such as divide and conquer, merging, solving by analogy etc in design of simple applications.	Analyze
CO4	Make use of functions, scoping and abstraction in development of simple applications.	Apply
CO5	Demonstrate mutability and higher order functions using file I/O and exception handling in python.	Apply

CO6	Practice software engineering principles like analysis, design, coding, testing and maintenance for the development of engineering applications using python programming	Analyze

The course is supported by a laboratory course "Computer Programming Lab" to provide rigorous hands on experience. The practical course includes the following ten experiments as listed in the Table 2.

Table 2- List of Experiments

1.	Simple Programs – Data types, operators and expressions
2.	Lists and Tuples Branching Programs
3.	Looping Programs
4.	String Programs
5.	Functions, Scope and Abstraction programs
6.	Structure Types and Mutablity Programs using Dictionary
7.	Higher Order Functions using Set
8.	File Handling and Exceptions
9.	Application of problem solving techniques Operations on Matrices
10.	Mini-project

Canvas is used as a learning Management System for engaging the students beyond class hours.

B. Incorporating Active Learning for in-class activities

A detailed course plan was developed and distributed to the learners, which includes details on content delivery method and formative assessment identified for each topic. Various strategies like Think Pair Share (TPS), Think Aloud Pair Problem Solving (TAPPS), Case Studies and Demonstrations were used in various sessions. The course plan could be found in the https://tinyurl.com/courseplan2018. Think-Pair-Share (TPS) is a collaborative learning strategy in which students work together to solve a problem or answer a question about an assigned reading. This technique requires students to

- think individually about a topic or answer to a question;
- share ideas with classmates.

Discussing an answer with a partner serves to maximize participation, focus attention and engage students in comprehending the reading material. A TPS activity in the session on looping constructs is presented below:

Think Phase: Identify an application in which "for" looping construct cannot be used and only "while" can be used.
Pair Phase: Discuss the correctness and appropriateness of the application.
Share Phase: Share the findings to the larger class group.

TAPPS was also practiced, where learners work in pairs. One of the learners will be solving a part of the problem by explaining the methodology and other will monitor and guide. The roles will be reversed in solving the second part of the problem.

Problem Statement:The program below finds the occurrences of a particular character in a string. Fill in the missing lines in the code (Learner – 1) and draw the flow chart (Learner – 2)

```
def check(string,ch):
if not string:
return 0
elif string[0]==ch:
return 1+check(string[1:],ch)
else:
    ……
string=raw_input("Enter string:")
ch=raw_input("Enter character to check:")
print("Count is:")
……
```

C. Problem Based Learning

Problem-based learning is a student-centered pedagogy in which students learn about a subject through the experience of solving an open-ended problem found in trigger material. Topics like File handling and Object Oriented Concepts were taught using Problem Based Learning. Heterogeneous groups have been formed and are every group is provided with a list of simple applications as listed below:

- Copy alternate lines of file a.txt to b.txt
- Count the number of words, lines and characters present in the file a.txt
- Change the all upper case characters present in the file a.txt into lower case and vice versa
- Search the given string (say hello) in the file a.txt. Count the number of occurrences of that string
- Search and replace the string in the file a.txt
- Assume the file contains Name, Contact Phone and Address in each line. Extract name and phone number of 5 different persons from the file. The data is separated by comma
- Read last n lines of a file.
- Read a file line by line and store it into a list.
- Find the longest words in a file
- Count the number of lines in a text file.
- Count the frequency of words in a file.

Learners explore the necessary python functions to complete the list of problems in a group. Instead of instructor explaining the entire library functions associated with files, learners explored the various functions according to the problem requirements. Learners were actively involved and engaged in completing many simple applications in a short period of time because of working in groups and learning from peers.

D. Problem Generation

Learners were encouraged to identify simple applications on their own to demonstrate each of the concepts like

Strings, Tuples, Lists etc., Uniqueness in selecting a problem of moderate complexity is ensured by sharing a google sheet among the learners to enter the problem statement. Before updating the details of selected problem, learners ensure that the same application is not selected by other students. Instructors gave guidance and feedback on the complexity of application chosen.

The codes developed by the learners were shared in the discussion forum of Canvas. The discussion forum facilitated learners to provide comments and suggestions for refinements on the works of others. Also, the discussion forum served as a good reference for learning since it includes a variety of applications for a topic of study. A snapshot of the discussion and responses is presented in Fig.1 and Fig. 2 respectively.

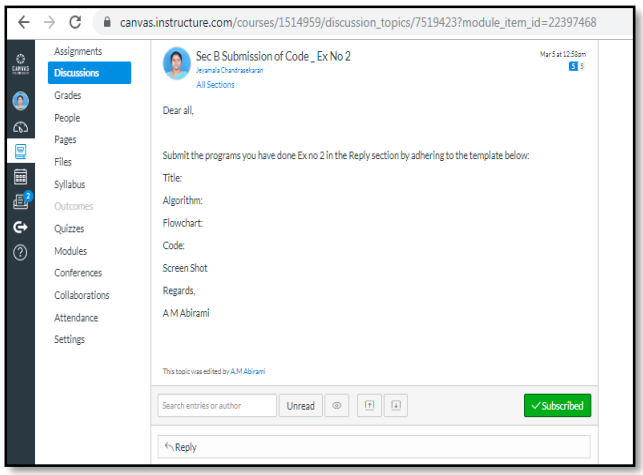


Fig. 1Initiation of discussion by the instructors

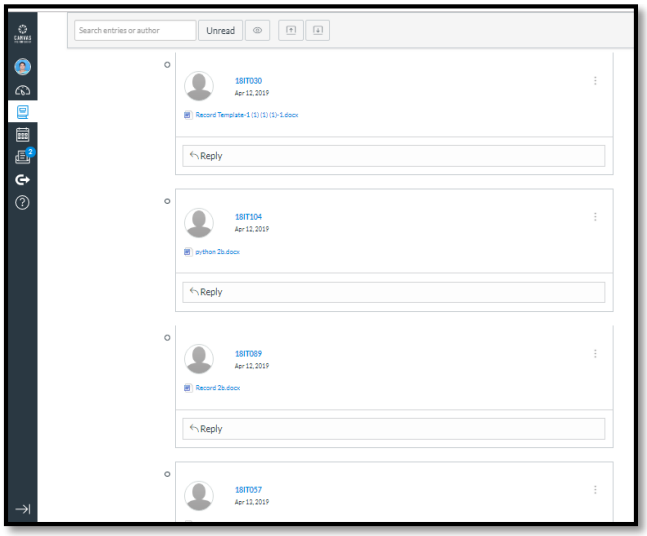


Fig. 2 Responses by the Learners

E. Exclusive parameters for grading of experiments

The parameters used for grading are designed based on the topic of study and were communicated to the learners well in advance. The parameters clearly explained the expectations and also helped in self evaluation. Learners could clearly identify the parameters to be worked upon for improving their grades. A sample for parameters used in grading the experiments on Strings and Functions in Python are depicted in Table 3 and Table 4 respectively.

Table 3 Parameters used for grading the activity on Functions

Parameters	Allotted Score	Actual Score
Complexity of the problem	5	
Use of function concepts (parameter passing, return values, naming of a function, etc)	5	
Use of different data types like numbers, strings, list, etc for parameter passing	5	
Demonstration of scoping of variables (local, global, nonlocal)	5	
Demonstration of Recursive functions	5	
Completion on time	5	
Viva (Online Test)	10	
Adherence to the template for documentation	10	
Total	**50**	

Table 4 Parameters used for grading the activity on Strings

Parameters	Allotted Score	Actual Score
Uniqueness of the Code (for the programs selected from the list provided by the instructor)	5	
Complexity and Uniqueness of the Programs (for the programs selected by the individual)	5	
Use of relevant string methods for solving problems	5	
Use of appropriate programming constructs like conditions, branching, looping etc., for solving problems	5	
Completion on time	10	
Viva (Online Test)	10	
Adherence to the template for documentation	10	
Total	**50**	

F. Online Quizzes for formative assessment

To provide reinforcement of learning, an online quiz was conducted using Canvas after completion of every topic. The quiz enabled the students to recall the concepts learnt and to correct their mistakes. The quizzes include questions at higher level of Blooms's taxonomy namely apply and analyse. Assessment items focussed on predicting the output of a code snippet and debugging an application. Student performance in quizzes served as an excellent tool for assessing student learning. A screen shot of quiz is presented in Fig. 3. The scores in the quizzes are counted towards the final grade and hence learners showed great interest in completion. The performance of the students in mid-term exams and End Semester Examinations was improved as a result of these online quizzes.

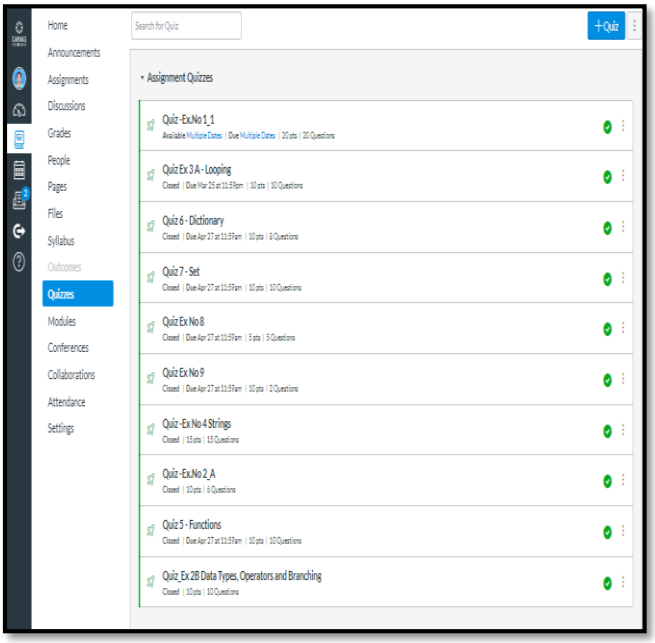


Fig. 3 Online quizzes for Formative Assessment

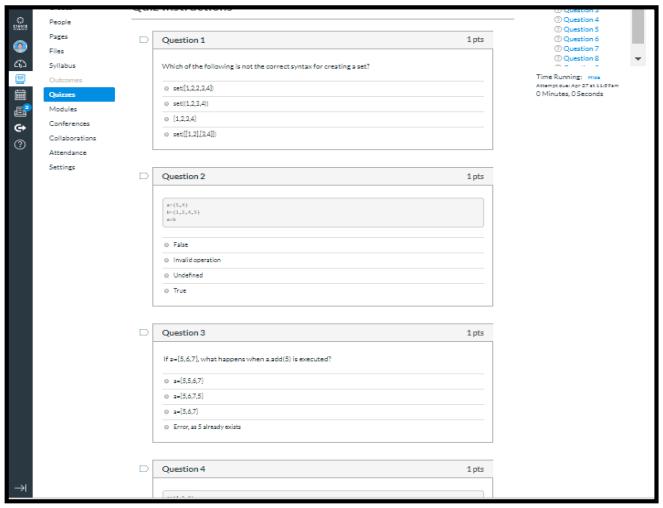


Fig. 4 Online quiz for the topic on "Sets" in python

G. Promoting Self Learning, Teamwork and Communication

To promote teamwork, communication skills and self learning ability, learners were grouped into teams and were made to explore any one of the libraries in Python. The topic of study was beyond the curriculum. Learners have to develop an application using the functions. The instructors supported in selection of libraries for exploration and in selection of application. The instructors also provided intermediate feedback and comments during the course of the mini project work. Rubric for grading has been circulated to convey the expected quality of the work and is presented in Table 5.

Figures 5 to 7 depict the snapshot of various applications developed by the learners.

Table 5 Rubrics for grading Mini project

Parameters	Excellent (5.0)	Good (3.0)	Satisfactory (2.0)	Unsatisfactory (0.0)
Correctness	Application runs and completes all required tasks Handles special cases Executes without errors	Application is complete in all aspects and competes most tasks appropriately Application fails to work for special cases	Individual modules produce expected output Application fails to handle errors due to integration	Individual modules do not execute due to errors. No integration of modules has been performed Incorrect results for most or all independent modules
Documentation	Clearly and effectively documented including descriptions of all variables. Specific purpose is noted for each function, control structure, input requirements, and output results.	Clearly documented including descriptions of all variables. Specific purpose is noted for each function and control structure	Basic documentation has been completed including descriptions of all variables. Purpose is noted for each function	No documentation included.
Coding Standards	Includes name, date, and assignment title. Excellent use of white space. Creatively organized work. Excellent use of variables (no global variables, unambiguous naming).	Includes name, date, and assignment title. Good use of white space. Organized work. Good use of variables (no global variables, unambiguous naming)	Completed between 70-80% of the requirements. Delivered on time, and in correct format).	Completed less than 70% of the requirements. Not delivered on time or not in correct format

Run time Efficiency	Executes without errors excellent user prompts, good use of symbols, spacing in output. Thorough and organized testing has been completed and output from test cases is included	Executes without errors. User prompts are understandable, minimum use of symbols or spacing in output. Thorough testing has been completed	Executes without errors. User prompts contain little information, poor design . Some testing has been completed	Does not execute due to errors. User prompts are misleading or non-existent. No testing has been completed.
Team work	Equal Participation and contribution Excellent support for each other Able to explain the logic of other modules	Contribution from few members Provide moderate explanation of other Modules Moderate Support for Team members	Contribution from one or two members in a group No clear idea on the work of others	No cooperation among team members No support for each other No idea on the work of other team members
Completed on time	Application is completed on time	Application is one day late	Application is three day late	Application is late by more than three days

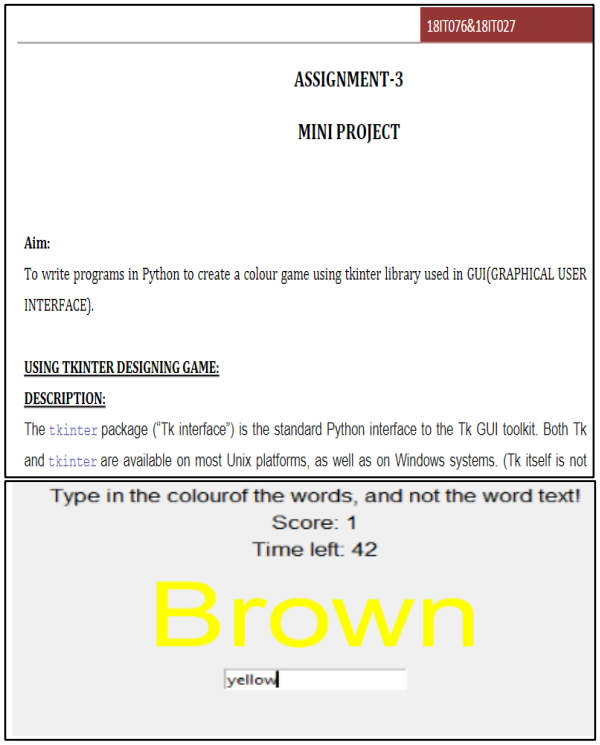
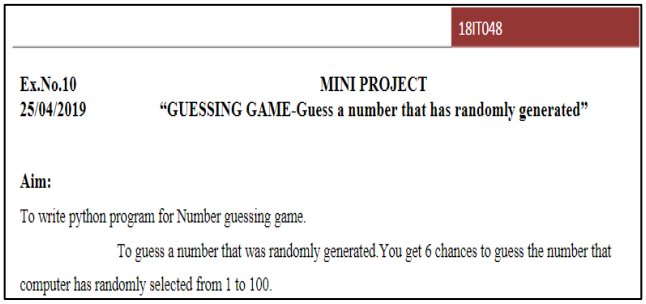


Fig. 5 Colour Game using Tkinter library

Fig. 6 Pattern printing using turtle



```
D:\>cd 18IT048

D:\18IT048>mini.py
Please Enter your name: malavika
Welcome to my Number game, malavika

I have selected a number between 1 to 100...
You have 6 chances to guess that number...
Enter your number: 23

malavika, My number is greater than your guessed number
you now have 5 chances left
Enter your number: 56

malavika, My number is less than your guessed number
you now have 4 chances left
Enter your number: 45

malavika, My number is less than your guessed number
you now have 3 chances left
Enter your number: 35

malavika, My number is less than your guessed number
you now have 2 chances left
Enter your number: 25

malavika, My number is greater than your guessed number
you now have 1 chances left
Enter your number: 30

malavika, My number is less than your guessed number
you now have 0 chances left
Sorry you lost the game!!
My number was = 28
Do you wish to play again?(y/n):
```

Figure 7 Number Guessing Game

H. Integration with MOOC

NPTEL course on "Joy of Computing using Python" was used as a supporting aid and reference for solving complex problems. Learners were insisted to register for the course and were instructed to complete the assignments. The score in the weekly assignments is counted towards the final grade. Registering for certification is made optional. Learners had a exposure on writing programs for simple gaming applications like

- Tic Tac Toe
- Rock Paper Scissors
- Snake and Ladders
- Hangman
- Checker Board
- Guess the Movie Name etc.,

During End Semester Examinations, learners were encouraged with bonus points, if they are able to complete such applications within the stipulated time.

5. Results and Discussion

A. Summative Assessment

The performance of students in the End Semester Examination in the theory and laboratory course has been analysed as an evidence of RQ1 and is depicted in Fig. 8 - 11.

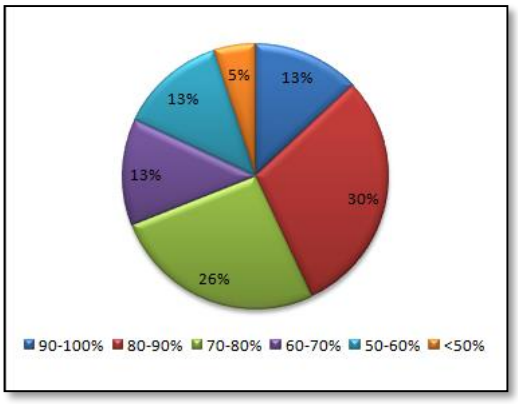


Fig. 8 Performances of Learners in Theory Examinations

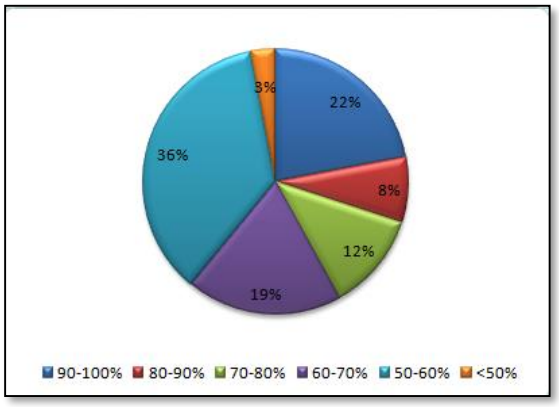


Fig. 9 Performance of Learners in Practical Examinations

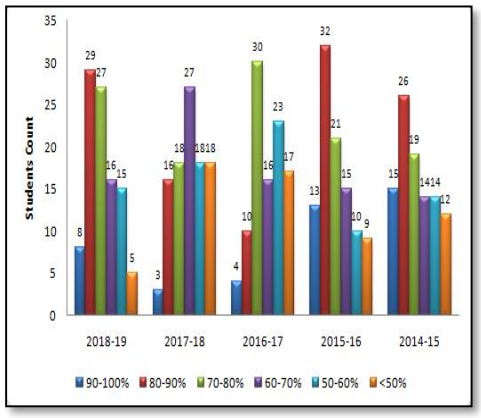


Fig. 10 Performance comparisons with previous batches for theory course

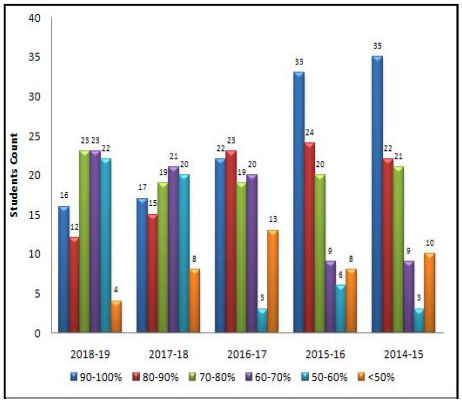


Fig. 11 Performance comparisons with previous batches for practical examination

It could be inferred from Figure 8, that the count of dropouts in the experimental group is found to be meagre (5% for theory examinations and 3% for practical examinations.. The count of failures has come down significantly and is the lowest, when compared with the results of previous batches. Majority of the students have scored greater than 60%. It could be inferred from Figure 10 and 11 that, the performance of the students in the

practical examinations for the experimental group under study has been significantly improved than the performance of the students admitted in the previous academic years. The improvement in performance can be used as strong evidence in support of RQ1 and hence it can be concluded that incorporation of active learning strategies results in enhancement of learning outcomes.

B. Course End Survey

The second part of RQ1 namely, the impact of active learning strategies on student engagement and feedback is assessed through course end surveys and mid-semester surveys. The satisfaction of students on the shift towards more hands on than the traditional lecturing method is measured through the corresponding research questions.

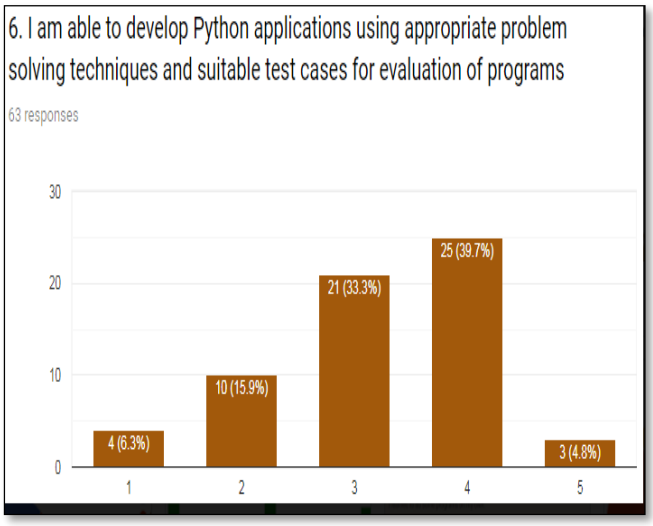


Fig. 12 Confidence level in Problem solving strategies

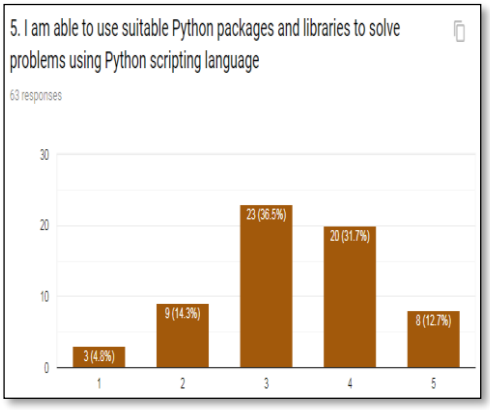


Fig. 13 Confidence level in using Python packages and libraries

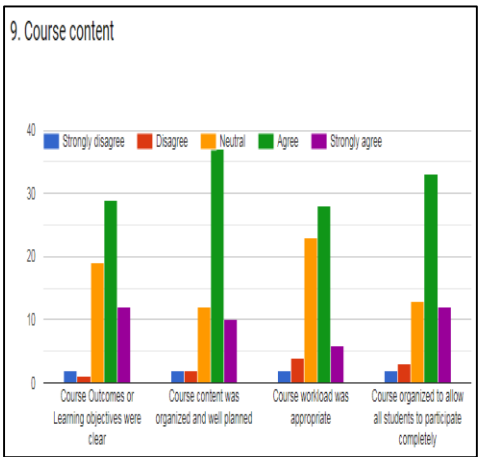


Fig. 14 Feedback on course content and delivery methods

Figures 12 and 13 depict the confidence level as measured through course end scale on an increasing likert scale of 1-5. About 77% of learners have demonstrated strong confidence in using various problem solving strategies and in developing test cases. About 80% of learners have shown strong confidence in using various python libraries for application development. From figure 14, it could be inferred that, more than % have strong agreement with structure and organization of content delivery methods. Qualitative feedback on the content delivery methods and assessment practices showed positive impact on student learning and engagement. Many of the learners preferred student centric methods with more emphasis on practical skill development and coding rather than traditional lectures. Few qualitative responses are depicted in Fig. 15.

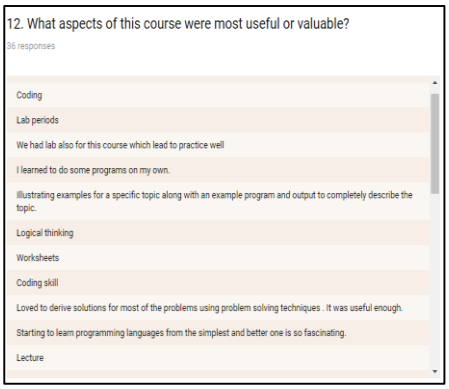


Fig. 15 Qualitative Feedback

C. Motivation for self learning beyond curriculum

Instructional strategies for the course have been carefully designed to promote self learning. In support of RQ2, the following evidences are collected to assess self learning :

- Count of Students participated in online courses beyond curriculum
- Ability of the students to take up projects of increased complexity.

. Incorporation of appropriate active learning strategies have motivated the students to participate and to earn certifications in online courses related to computer

programming. The statistics in participation of online courses is presented in Fig. 16.

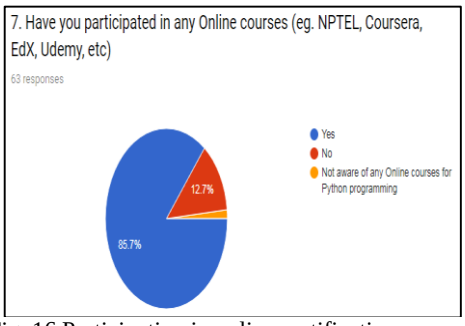


Fig. 16 Participation in online certification courses

The effect of successful learning can be measured by the count of students taking up the course to the next advanced level of study. It could be inferred from Figure 16 that, more than 85% of the learners have shown interest in acquiring online certifications in Computer Programming from NPTEL, Coursera, Udemy etc., Section 4G highlights few mini projects carried out which involves exploration of suitable libraries by the learners themselves. The quality of the project works demonstrate that the self learning ability of the learners has been improved significantly.

6. Conclusion

The experimental work aims at systematic integration of active learning strategies for the freshman course on problem solving using computers. Detailed investigation on the performance of the learners clearly indicates that active learning strategies have a significant impact in increasing the learning outcomes. It can be observed from the feedback of the learners that the satisfaction index on the content delivery methods is high. Learners were completely engaged in practical coding and enjoyed teamwork. The course has provided the necessary foundation for certification in online courses and motivated the learners for participation in programming and debugging contests. Future extension of the research work is to design a personalized learning framework for the course on introductory computer programming. The frame work has to include multimodal content delivery and learners will be provided with a preference to select the learning path based on their learning style.

References

1. AdalbertG. S. R ,Jignesh M. P, and Richard H (2018). Is More Active Always Better for Teaching Introductory Programming? Proceedings of IEEE Learning and Teaching in Computing and Engineering (LaTICE).
2. Astrachan, O. L., Duvall, R. C., Forbes, J. and Rodger, S. H., (2002) Active Learning in Small to Large Courses, Proceedings of the 32nd ASEE/IEEE Frontiers in Education Conference.
3. Briggs, T (2005), Techniques for active learning in CS courses, Journal of Computing Sciences in Colleges, 21(2), 156-165.
4. Gottfried, Byron S. (1997) Teaching computer programming effectively using active learning." Proceedings of American Society of Engineering Education Annual Conference.
5. Bullard,Felder, R. M and Raubenheimer, M. (2008) Effects of ActiveLearning on Student Performance and Retention. ASEE AnnualConference Proceedings, ASEE, June 2008
6. Duffany, J. L. (2017) Application of Active Learning Techniques to the Teaching of Introductory Programming,IEEERevistaIberoamericana de Tecnologias del Aprendizaje, 12(1), 62-69.
7. M. Ebert, (2017), Increase active learning in programming courses, Proceedings of the IEEE Global Engineering Education Conference (EDUCON), 848-851.doi: 10.1109/EDUCON.2017.7942946
8. Felder, R. M and Brent R. (2009) Active Learning: An Introduction. ASQHigher Education Brief, 2(4), August 2009
9. Hake, R. R. (1998) Interactive-engagement versus traditional methods:A six-thousand-student survey of mechanics test data for introductory physics courses, American journal of Physics, 66(64).
10. Martínez, C.L., & Muñoz, M. (2014). ADPT: An active learning method for a programming lab course.Proceedings of the 10th International CDIO Conference
11. Zimudzi, E. (2012). Active learning for problem solving in programming in a computer studies method course, Academic Research International, 3 (2), 284-292.