Journal of Engineering Education Transformations ,
Volume 33 , No. 3, January 2020, ISSN 2349-2473, eISSN 2394-1707

An Innovative Approach for Learning and Evaluating Programming-Oriented Courses

Dr. Paresh Tanna[1] , Dr. Nirav Bhatt[2], Sweta Patel[3]
[1]School of Engineering, RK University, Gujarat, India
[2]School of Engineering, RK University, Gujarat, India
[3]School of Engineering, RK University, Gujarat, India
[1]paresh.tanna@rku.ac.in
[2]nirav.bhatt@rku.ac.in
[3]sweta.patel@rku.ac.in

Abstract: As many of the students are facing problems in programming languages. Most exciting task to teach such languages is with the development of logical skills of the students while learning programming language courses. Also, most of the programming language courses are evaluated with only problem definitions or remember/understanding level basic questions. Students may be very less experienced to write full solution/missing/incorrect code for the given task. Secondly, by writing solution from scratch is not the only way to work with IT industry. There are many more situations where students may need to perform equally. Such situations may be found based on industry to industry work culture, type of requirements, coding style etc. Best technique i.e. troubleshooting & debugging that can be used to build logical skills and cover up said industry demands.

Keywords: Programming Languages, Logical Skills, Tec hnic al Skills, IT Industr y Demands, Troubleshooting & Debugging, Innovative Assessment

Dr. Paresh Tanna
RK University, Gujarat, India
paresh.tanna@rku.ac.in

1. Introduction

The students taking admission in computer science/engineering are from various backgrounds. About 70-80% of them will not be very conversant in English language in our region. Many of them are come up with traditional pen paper method-based examination. So naturally mugged up was the base for them to clear any exam. To bring them out from such tradition, something needs to be done for computer science/engineering students so that students can work properly in this field after completion of their degree. In fact, logical skills and technical skills matter a lot for computer science/engineering students. Such highly demanded skills can be developed with the programming languages courses (Sharma, A., 2019). So, here the role of faculty is more important to teach and assess the logical/technical skills based on concerned programming language courses' requirements in the IT industry.

2. Literature Review

RoboProf (developed in 1999) is a web-based instructing framework dependent on the World Wide Web innovation. RoboProf shows syntax and structure of C++ programming language to students and assesses the activities students comprehend.

Limitation and Justification: This is only limited to syntax and structure level test. While proposed method focuses on many parameters like syntax

errors, logical errors, compilation and output generation, writing comments for important logic, rearranging the code to get the said output, analysis and evaluation of existing code, writing full module, etc.

Nagappan et al explored the utilization of pair programming in their early on processing course and detailed that pair programmers were increasingly independent, for the most part performed better on undertakings and tests and were bound to get an evaluation of C or higher than their independent partners (Nagappan, 2003).

Limitation and Justification: This is only limited to C programming language. Also, it considers pair-based evaluation approach. While proposed method is suitable for any programming-oriented languages along with many parameters (as mentioned in first paragraph of this section).

Mooshak tool was created in 2003 by Leal and Silva. Mooshak is web-based programming challenging tool. It helps in the behaviour, answer assessment, result affirmation, and feedback. The languages bolstered naturally are C, Pascal, C++, and Java.

Limitation and Justification: This is only limited to full program testing which is sometimes not feasible to assess the students overall practical skills. While proposed method is suitable for any programming-oriented languages along with many parameters (as mentioned in first paragraph of this section).

In 2003, paper by Zhenming et al. and later reached out by Zhang et al. in 2006, tool was created to quantify working aptitudes of the students and their proficiency in utilizing tools or applications for programming languages or courses.

Limitation and Justification: This approach is limited to only basic programming features compare to proposed approach.

Collaboration is an active learning strategy that has been related to increased achievement in engineering students (Baylor, A. L., 2005).

BOSS (http://www.boss.org.uk/) is an online submission framework, which gives highlights to course the board and automated testing. It could assess C, C++, and Java programs. It utilizes JUnit for testing.

Limitation and Justification: This approach compares only output of the whole program. While proposed method is suitable for many parameters (as mentioned in first paragraph of this section).

Static methodology checks and investigates the source code without executing the program (Ala-Mutka, 2005) and being utilized to assess the programming style, linguistic structure and semantic error, programming metric examination, basic closeness examination, keyword detector, plagiarised content finding and furthermore outline analysis.

Limitation and Justification: This approach is limited to whole program in concern to different structure and syntax only. This approach does not cover the student thinking on correction of code or evaluation and use of code which is covered in proposed approach.

The traditional way to deal with evaluating student comprehension and capacities includes a mix of programming assignments and written tests. Commonly, teachers give out-of-classroom assignments planned to test student program writing capacities, and they give in-class tests to test how a lot of students comprehend of the ideas. These tests are taken and utilized as an extra keep an eye on language syntax and logical structure (Smith, P. P., 2005).

Limitation and Justification: The issue with this methodology is that it doesn't portray the two aptitudes well overall. This is on the grounds that, from one perspective, the teacher can never be sure precisely who composed the programming assignments, and then again because programming angles are blended in with the written tests. For instance, it's normal to ask questions about the idea of control structures directly alongside questions concerning control structures in the language of intrigue. This makes it hard to tell whether the students know either.

While proposed approach is totally practical along with many types of programming skills testing questions which are ranging from remembering to creation level of blooms (as mentioned in first paragraph of this section).

Another serious issue concerns the setting an environment wherein out-of-class programming

assignments are finished. It is enticing, and simple, for students to get help with writing these programs. In fact, plagiarism and cheating are genuinely regular in such circumstances. Thus, since the genuine developer of a given program is in question, the last application may not be at all indicative of the skills level of the students who hands it in (Smith, P. P., 2005).

Limitation and Justification: In this approach chances of plagiarism and cheating are genuinely common. While proposed approach overcomes these with its paper style and assessment pattern along with regular practicing practical assignments.

Douce et al., in 2005, created Automated System for Assessment of Programming (ASAP) code which fits into the e-learning structure as an assessment instrument.

Limitation and Justification: It is limited to C programming only and also it checks a single program at all. While proposed approach have many ways to check and acknowledge about students' practical skills.

Outside-of-class programming assignments are given to assess the students dynamic and increment in trouble as the semester continues. The reason for existing is, obviously, to offer students the chance to execute strategies of programming clarified in class, and to make it feasible for them to practice these aptitudes in an assortment of contexts. Likewise, they fortify the many programming concepts instructed during classroom session (Smith, P. P., 2005).

Limitation and Justification: As we execute the methodology here, assignments might be done in gatherings, and students can use any tools they wish gave just that they recognize such outside help. Since these assignments contribute negligibly to the final course grade for a student, there is less motivation to cheat to complete them effectively.

While proposed approach is done only in controlled environment under the supervision of faculty member.

Mandal et al. examined, in their paper distributed in 2006, the design of an automatic program assessment framework. This framework can deal with just C programs. Their methodology is to perform white box test, rather than, black box or dark box testing. SQLify (2007) is to help students to learn SQL. It encouraged composing test queries against databases; get quick criticism which is more instructive than what can be offered by DBMS.

Limitation and Justification: These approaches are only limited to C programming language and SQL language respectively. Also, it checks only whole program/procedure at all. While proposed approach have many ways to check and acknowledge about students' practical skills.

In 2008, Farrow and King utilized a web-based programming assessment to evaluate students who had learned Java for two terms. It utilizes BLUEJ and JUnit programming tool.

Limitation and Justification: This approach can be adopted to check only java programming language and it is limited to whole program only. While proposed approach is more suitable for many programming languages along with many parameters (as mentioned in first paragraph of this section).

In 2009, García-Mateos and Fernández-Alemán and Montoya-Dato et al. utilized Mooshak for compute programming-oriented courses for assessment.

Limitation and Justification: This approach can be adopted to check only C & C++ programming languages and it is limited to whole program only. While proposed approach is more suitable for many programming languages along with many parameters (as mentioned in first paragraph of this section).

In 2010, Zhang and Ke also suggested a design for SQL Paperless Examination System.

Limitation and Justification: It is limited to SQL only with basic predefined queries and procedures only in contract to proposed approach which is useful with type of skills tests.

The conventional evaluation approach, in which one single written assessment tallies toward a student's final score, never again satisfies new needs of programming language courses. In view of a peer coding process model, they built up an online evaluation framework called EduPCR and utilized a novel way to deal with assessment of the learning of programming-oriented courses. Utilizing this methodology, students peer review programs written

by different students, share thoughts and make proposals to accomplish a target of cooperative and intelligent learning (Wang, Y., 2012).

Limitation and Justification: This approach uses the peer assessment pattern this may result to biased score. While proposed approach is fully educator centric.

In the classroom, the teaching-learning process is the teacher's responsibility. Interaction among students in programming courses happens when they face such programming problems as debugging errors (Rienovita, 2014).

Limitation and Justification: This involves the practical skills improvement in rare number of students and also based on student code writing pattern. Many times, this excludes the industry need while the is covered in proposed approach.

Academicians and IT experts give greater importance for mastering many programming-oriented languages. Learning numerous language syntax and constructs won't contribute much in building critical thinking abilities. For some random assignment, students ought to be fit for fathoming it by utilizing any of the programming languages. More significantly, students should develop a specialty of improving the performance of the application by applying principal and propelled ideas of programming languages (Husain, M., 2015).

Justification: Referring above finding, proposed approach is built in such a way that to cove all the relevant practical skills to be covered while teaching and assessment both.

Authors utilize the C++ programming language, expanded with an illustrations library and some semantic gadgets we have created. They have discovered that our increases are helpful in clarifying many programming concepts, for example, recursion, and obviously empower perception and graphical association. Notwithstanding showing the language structure of C++ they show how fascinating programs from science, engineering, and different research activities can be created with generally little exertion (Ranade, A., 2016).

Limitation and Justification: This approach only focuses on basic concepts coverage of C++ language which is far away compare to proposed approach (as mentioned in first paragraph of this section).

Another methodology as hidden part in the game condition, is to draw in the understudy in the framework, making him/her keen on the framework, inspiring him/her to advance in the game, beating snags, and thusly causing him/her the will to program (de Jesus Gomes, 2017).

Limitation and Justification: For assessing students' practical skills, this approach requires hard work from subject faculty in terms of designing a game which also demands a lot of technical skills from subject faculty. This approach is practically not possible for almost faculty. While proposed approach helps subject faculty with lesser time but for more improvement of students' practical skills.

"Code Factory" is an inviting and natural condition, where students can investigate centre programming ideas in a dynamic, outwardly rich, and spurring way. Code Factory is separated into two centre intuitive conditions, the "code development condition" and the "game condition". These contrast in presentation, purpose and UI, considering the data and apparatuses to be displayed to the students just when fundamental and making an emblematic change between the two distinct sorts of ongoing interaction (de Jesus Gomes, 2017).

Limitation and Justification: This approach is only limited to testing of few concepts of a stipulated programming language in contract to proposed approach which is helpful to align regular work as per IT industry needs.

A consistent assessment procedure for a computer programming course bolstered by a programmed evaluation device, applied to the functional programming practices performed by the students. The connection between the students and the assessment tool was contemplated through quantitative investigations. Specifically, the programs proposed by the students (computer programs) were broke down utilizing the decisions given by the programmed assessment tool: correct solutions or incorrect solutions. On account of incorrect solutions, the kinds of programming errors were considered (Restrepo-Calle, F., 2019).

Limitation and Justification: This approach only results in correct or incorrect program. This also demands whole program to be written for assessment.

This is also limited to C programming language. While proposed method focuses on many parameters like syntax errors, logical errors, compilation and output generation, writing comments for important logic, rearranging the code to get the said output, analysis and evaluation of existing code, writing full module, etc.

The procedure utilized in all the apparatuses for assessment are limited to whole programs and some are only limited to specific programming language. An increasingly effective encompassing methodology or approach is required for assessing the student programs . Aside from identified limitations/findings, the program quality and semantic comparability of program with many parameters should be built up for programming or technical skills development. Adhere to this, proposed approach is designed for all type of programming languages with different questions format.

3. Methodology

As per the traditional pen paper approach, students are not evaluated based on their skills, but they are evaluated based on mugging up method i.e. writing answers of remembering/understanding based questions in the answer sheet. To overcome the problems in traditional methods and to improve the students logical and technical skills, one of the important methods is evaluation based on troubleshooting and debugging. This troubleshooting & debugging methodology deals with assessment of the student's ability to analyse, evaluate and apply the technical knowledge of programming language. This methodology will increase the student 's ability and interest to perform the desired operation. This methodology will be based on continuous evaluation: Lab experiments will reflect all the categories of assessment methods and faculty will evaluate lab experiments at regular intervals to check the progress of students. This troubleshooting & debugging examination is conducted as regular midterm internal examination.

A. Execution of troubleshooting & debugging method

This method execution will be based on following:

- Exam is conducted in computer laboratory using Canvas Platform.
- Faculty has created the quiz by adding essay type questions from different categories like error identification & correction, writing missing code statement, code correction for logic errors, predict output of code, code comments, correct the output, arrange the code, etc.
- Students had to troubleshoot the given questions as per the provided instructions and re-write the correct line of code in the provided answer area.
- Total 20 questions each of 2 marks were provided to be solved in 90 minutes of duration.
- At the end, faculty assessed each question using dedicated rubric.

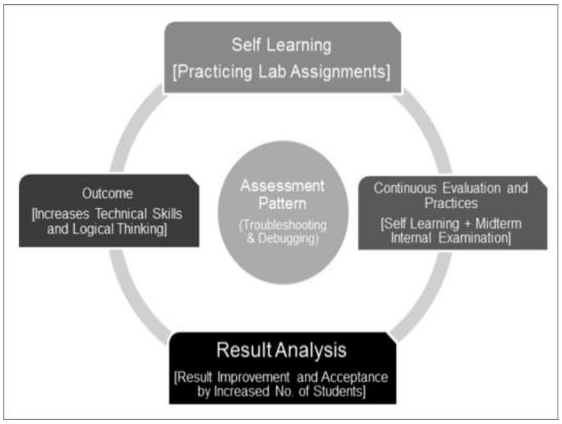

Fig. 1: Execution Cycle for Troubleshooting and Debugging Approach

Here, Fig. 1 highlights the execution cycle for troubleshooting and debugging approach. First students have been assigned practicing lab assignment before taking exams. During mock test, students has been guided about the method and practiced a lot about innovative method. Then student examination has been taken and analysis on performance with score has been discussed. Identifying and sharing mistakes in this exam, may guide students about a lot for improvement in technical skills and logical thinking.

4. Process of Assessment & Documentation (Assessment Design)

Each question may contain 'n' number of errors depending on the level of question and its weightage. Students apply the knowledge to identify syntax error, logical error, missing code line, predict the output of given code, correct the output, arrange the code etc. Course faculty assessed each question using the

discussed rubrics in this paper.

A. Importance of proposed methodology (Why)

For logical/technical skills development, it is very important for the faculty to check whether students are learning or not, i.e. “We are teaching. Are they learning?” quoted by IUCEE. Is this happening with your class/students? To work on this, proposed methodology is important for the following reasons:

- To sharpen the memory of students by syntactical error and logical error checking techniques
- Student’s problem analysing and code writing skills
- To improve analytical ability through missing code
- To increase the predicting power by giving them a code to judge the output
- To evaluate students regular programming comments practices
- To apply their fundamental logic by arranging different coding statements

B. Purpose of Students/Faculty involvement (Who)

Student's problem analysing and code writing skills will be evaluated based on this methodology.

Course faculties do continuous evaluation during lab hours with the help of lab assignments and continuous midterm internal examinations.

C. Execution Schedule (When)

In order to evaluate learning on a continuous basis, lab assignments are evaluated regularly as well as proposed method is conducted during regular midterm internal examination.

D. Location of Examination (Where)

Online examination is conducted and assessed using learning management platforms i.e. Canvas.

E. Advantages of Proposed Method over Existing Method

Compared to traditional pen paper examination, we can find following benefits:

· The self-learning approach of students increases through solving missing code, syntax and logical error checking techniques.

· The student’s memory is sharpened by remembering the syntax of languages, in-built functions and bugs occurred in past and its solution

· The technical knowledge of students is enhanced by rearranging the code and predicting the output rather than writing theoretical traditional answers.

F. Expected Challenges of Proposed Method

Due respect to traditional method, we can find some challenges in this proposed method:

· Students may submit answers in different formats that will make assessment difficult.

· The proposed method does not include code optimization techniques.

· Paper setting should cover all the aspects of the whole syllabus.

· Students must be familiarized to solve such kind of problems during lab sessions.

G. Sample Questions with Logical/Technical Skills Development

Students are learning different programming languages in classroom or lab. Faculty is not known whether students are learning or not. If they are learning, then how they are learning its again a question raised in concern to mugged up. Many students follow last day learning i.e. last day before internal or external examination. To check students learning in regular way proposed method is the best way to add debugging and troubleshooting kind of questions into tutorials and internal/external examination for programming courses. Some examples are illustrated in below part along with blooms' levels alignment with such type of questions. Questions no. 1 to 3 are discussed for Java

programming in below portion.

Q-1 Write missing statement(s) for the following to generate desired output.

```
class X extends Thread
{
    public static void main(String args[])
    {
                    //missing statement(s)
    }
    public void run() {
    System.out.println("from X runnable");
    }
}
```

output:

from X runnable

This type of question will check the debugging and troubleshooting knowledge for thread problem by finding missing code into the given Java code.

Q-2 Rearrange the following code to get the desired output.

```
{
{
    double c = Double.parseDouble("9");
    System.out.println(x);
    System.out.println(c);
    int x =Integer.parseInt("5");
  }
}
public static void main(String args[])
public class Test
```

Output:

9.0

5

Q-3 Predict the output and write comments for important logic.

```
class Test
{
    public static void main(String [] args)
    {
    int x=20;
 String sup = (x < 5) ? "small" : (x < 25)? "medium" :
"big";
    System.out.println(sup);
    }
}
```

As we know writing output for the program, requires lots of debugging knowledge. Faculty can take the help of such questions to check the debugging knowledge of students for different topics.

Q-4 Analyse and compile the given package1 and import the same package1 into another package i.e. package2 in order to utilize the methods developed in package1.

```
package package1;
public class MyMathCalc
{
      public static long findExponent(int base, int
exponent)
   {
      long result = 1;
```

```
    while (exponent != 0)
    {
        result *= base;
        --exponent;
    }
    return result;
  }
}
```

This type of question helps student to analyse the existing code and increase their thinking capability to utilize the same for developing reusable code.

Q-5. Identify the runtime bugs by testing the below given code snippet and rewrite the correct code.

```
public class MyClass {
 public static void main(String[ ] args) {
  int[] myNumbers = {1, 2, 3};
  System.out.println(myNumbers[10]);
 // runtime exception will be generated here!
}
```

This type of question requires higher level thinking. Thus, checking the student's debugging and error solving capabilities will insists students to think about the errors/exceptions, evaluate the expressions and find appropriate solution with exception handling.

Q-6. Create new methods or modules to solve the problem of inserting, updating and deleting a record from a database table with more efficiency. (Hint: Create methods using JDBC for different required operations and use Java Swing Controls for UI design.)

Rather than solving the bugs or evaluating the existing code, such type of questions helps to develop student's creativity by providing the task i.e. develop any programming module from scratch.

Considering the blooms level, such type of assessment method covers all the aspects ranging from remembering, understanding, applying, analysing, evaluating to creating the code modules as compared to writing the same on paper i.e. written pen paper exam. This increases the technical knowledge as well as the confidence of developing and solving the real-time application problems.

By concerning the comments importance in the IT industry, where people work in a group; common project is being allocated to many programmers, adding comments to methods or logical statements helps other programmers to understand the existing code in a better way. Also, editing the commented code looks easy, efficient and faster by other programmers with clear understanding

Adding such type of question along with output and comment, help the faculty to mould students as per industry standard practices.

A. Question Format Alignment with Blooms Level (Dontham, 2016)

1. Remembering level: Q-1 is generally being asked in traditional way like: Explain Thread object creation and thread starting process with example.

2. Understanding level: Q-2 is generally being asked in traditional way as Illustrate use of int and double data types. Also explain process to convert string value into int value.

3. Apply level: Questions 1,2 and 3 are examples of this level i.e. missing code, rearrange the code, write output of the given code etc.

   Questions related to syntax and logical errors finding, can also be added as debugging and troubleshooting format at blooms 3rd level.

   Referring Questions 4 to 6, it will justify as appropriate examples for how higher order thinking skills that can be used to assess by proposed approach.

4. Analyse level: Question 4 implies an analytical question, where provided code is divided into several parts and students are asked to organize/utilize (by analyzing the given code) the same to achieve an objective. The critical component and unimportant component are also identified.

5. Evaluate level: Question 5, refers to check the evaluating capability, where testing is performed to determine whether the code satisfies the given task objective and able to suggest/correct or produce better code by performing the assigned task.

6. Create level: Question 6, refers to the highest thinking order where students are asked to develop modules/methods from scratch using individual's technical and logical skills which in turn helps to assess the correct order of creation level of blooms.

Hence, by considering the above all the sample questions, troubleshooting and debugging approach covers all the facets one needs to view while assessing the student's knowledge, especially for programming courses.

5. Result Analysis

Here, to analyse the impact of troubleshooting and debugging method, the result of TCIE2 (Theory Continuous Internal Examination 2) of MCA 4th Semester students is considered in below table with different parameters (Ganesh, 2018). Result of TCIE1 of the same group was like around 35% students had secured more than above average marks with traditional method-based examination while comparing the same with TCIE2 which was taken with troubleshooting and debugging method is like around 69% students had secured more than above average marks.

In addition, topics covered for TCIE2 was advanced

Table 1 : TCIE1 & TCIE2 Result Analysis Comparison

Result Analysis (Academic Year 2018-19 MCA 4th Semester)					
Level	Grade	Grade Points	Marks in Percentage	% Students TCIE2 (Troubleshooting & Debugging)	% Students TCIE1 (Traditional)
Outstanding	A+	10	Marks ≥ 90%	15.38	3.85
Excellent	A	9	80% ≤ Marks < 90%	23.08	7.69
Very Good	B+	8	70% ≤ Marks < 80%	23.08	11.54
Good	B	7	60% ≤ Marks < 70%	7.69	11.54
Above Average	C+	6	50% ≤ Marks < 60%	19.23	7.69
Average	C	5	45% ≤ Marks < 50%	3.85	19.23
Poor	D	4	40% ≤ Marks < 45%	0.00	26.92
Fail	F	0	Marks < 40%	7.69	11.54

compared to TCIE1. This comparison shows that students are more familiar and interested to solve logical/technical problems rather than mugged up of theory concepts and writing the answer in traditional pen paper method. Regular practices with troubleshooting and debugging method will improve the logical/technical skills as well as will inspire more stude nts to take admission in computer science/engineering programs.

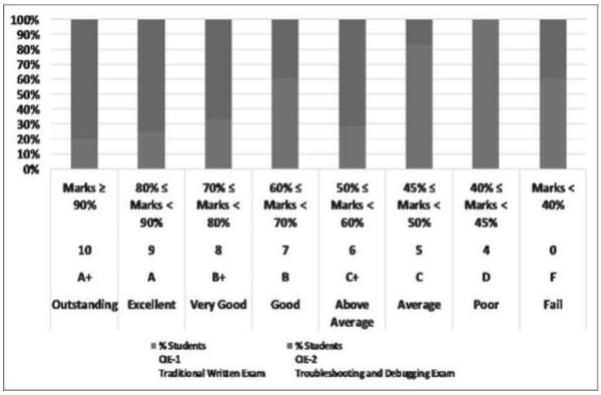

Fig. 2: TCIE1 & TCIE2 Result Analysis Comparison

Note: TCIE means Theory Continuous Internal Examination

As shown in Fig. 2 Result of TCIE1 of the same group was like around 35% students had secured more than above average marks with traditional method-based examination while comparing the same with TCIE2 which was taken with troubleshooting and debugging method is like around 69% students had secured more than above average marks.

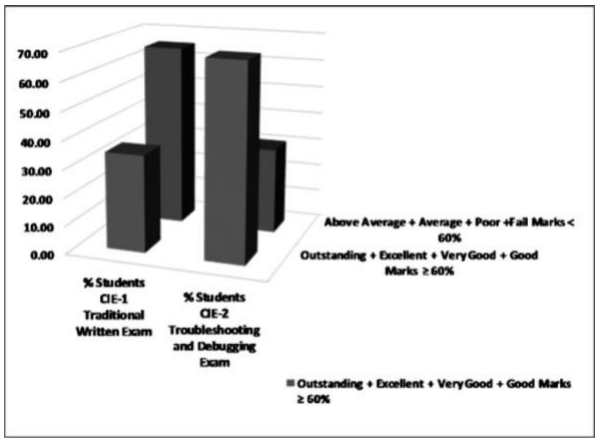

Fig. 3 :TCIE1 & TCIE2 Result Analysis Comparison with Grouping (>=60% Students Group & <60% Students Group)

Control group and experimental groups are set up to evaluate the impact of proposed approach on students learning as following. The control group i.e. traditional pen paper approach (TCIE1 Exam Assessment i.e. given in last column) and experimental group (TCIE2 Exam Assessment i.e. given in last second column) i.e. troubleshooting/debugging approach are compared against each other in this experiment.

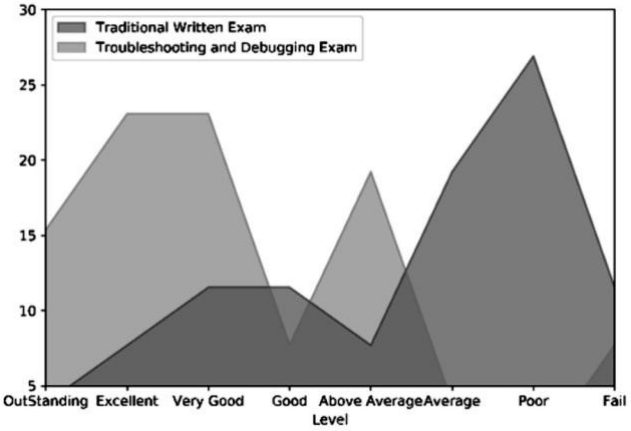

Fig. 4 : Comparison of Traditional and Innovative Exam Assessment

Fig. 4 is also shown in different format i.e. given in Fig. 3. It also highlights increasing number of students with grades (Outstanding + Excellent + Very Good + Good) supports that student knowledge increased compare to traditional exam approach.

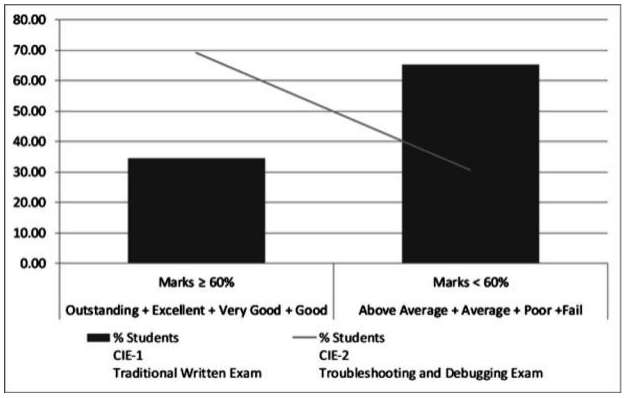

Fig. 5 : TCIE1 & TCIE2 Result Analysis Comparison with Grouping Variations (>=60% Students Group & <60% Students Group)

The differences between the two groups are that the independent variable i.e. practical knowledge is changed in the experimental group. The independent variable is "controlled" or held constant in the control group i.e. by writing question paper answers in traditional way may inspire students to mug up a lot for programming subjects. Considering experiment group, students are eager to solve their problems in labs. During placement drive, students are very much cure with technical tests to solve programming-oriented questions.

Fig. 5 is also shown in different format i.e. in Fig. 3. It also highlights the same that the proposed approach shows more improvement in result i.e. increasing percentage of students with >=60% score (above average score).

The hypothesis testing using paired sample t-test was used for this study. The outcomes are expected to get higher mean value for troubleshooting and debugging approach than traditional written approach of assessment. The null hypothesis H0: There is no difference in mean for troubleshooting and debugging approach than traditional written approach. The alternative hypothesis H1: There is a difference in mean for troubleshooting and debugging approach than traditional written approach.

The exam papers were evaluated by the faculty and score were classified into grade according to RK University's final examination standard. Then, the descriptive analysis was used to elaborate the mean and the standard deviation of the score. Since the percentage marks for traditional and troubleshooting & debugging exams were collected from the same students, the paired t-test was used to test the significant difference for both exams' score. The paired t-test was conducted to test the hypothesis.

The data collected from both the exams were tabulated in order to investigate the effectiveness of the proposed approach. Table 1 depicts the overall score for both exams in which the students' performance was classified into eight categories, fail, poor, average, above average, good, very good, excellent, and outstanding. From Table 1, the students' results are seen improving remarkably after following the proposed approach.

Level	Traditional Written Approach (%)		Troubleshooting and Debugging Approach (%)	
	Mean	Standard Deviation	Mean	Standard Deviation
Outstanding + Excellent + Very Good + Good	8.65	3.68	17.31	7.36
Above Average + Average + Poor + Fail	16.35	8.53	7.69	8.31

Table : 2 The Paired Samples Statistics of Traditional and Troubleshooting & Debugging Approach

Table 2 shows the paired samples statistics of traditional written approach and proposed approach.

The lowest mean with the mean value 8.65% is found in Traditional Written Approach for level 1 (Outstanding + Excellent + Very Good + Good) grades and this implied that students are very weak to perform in Traditional Written exam for programming subject. On the other hand, proposed approach-based exam with the mean value of 17.31% indicated that students have better understanding in solving practical problem beforehand. Yet the result shows that there is about 50% of improvement with proposed approach for higher grading. The figure revealed that the proposed approach had intensifying the students' capability in solving the programming problems. In traditional written approach, the standard deviation is quite dispersed compared to proposed approach for level - 1 (Outstanding + Excellent + Very Good + Good) grades. While on the reverse side both mean and standard deviations are higher for the traditional written approach for level - 2 (Above Average + Average + Poor + Fail) grades and this implied that students are very weak to perform in Traditional Written exam for programming subject.

Level	Paired Differences (%)		t-test result (2-tailed)
	Mean	Standard Deviation	
Outstanding + Excellent + Very Good + Good	8.66	3.68	0.135

Table : 3 Results of Paired Samples T-Test of Traditional and Troubleshooting & Debugging Approach

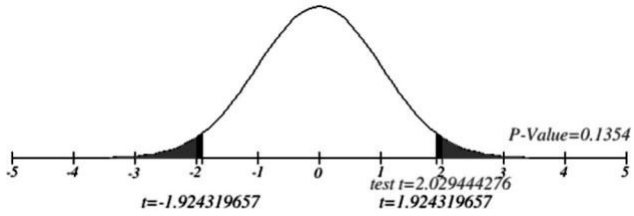

Fig. 6: Results of Paired Samples t-Test over Level – 1 Grades (Outstanding + Excellent + Very Good + Good)

Table 3 and Fig. 6 show the results of paired t-test for level-1. At 15% significance level, all null hypotheses are rejected and t-test result i.e. P-value indicates that with proposed approach (troubleshooting and debugging) results has been improved drastically compare to traditional written approach, thus it can be concluded that there is a statistically significant difference between mean mark of the traditional written and proposed approaches. Hence, it is strongly evident that the proposed approach is able to enhance the students' practical knowledge or skills effectively.

As per the website (http://placement.freshersworld .com/placement-papers) which is widely used by students for placement preparation, most of the IT companies are focuses on technical part for problem solving with troubleshooting/debugging method. In fact, none of the companies, has asked a question like “explain”, “list out”, “describe” etc, kind of questions in their technical tests. Many MNC companies only follow technical tests before interview round. In such case students must justify their knowledge with proper technical skills. Proposed approach will help students to perform well during their placement drive.

Rubrics with Discussion:

The assessment rubrics will be based on implementation with proper techniques, sequence, syntax and comments done by students with different criteria:

I. Student code shows complete (100%) implementation of the given task. All steps are in correct sequence. Code implementation is without any syntax errors. All comments are put up properly. In this case the student will be able to secure 100% marks (2 Marks) for the assigned task/question.

II. Student code shows complete/partial (75%-100%) implementation of the given task. All steps are in the correct sequence or few steps are not in correct sequence. Comments are not put up properly. In this case the student will be able to secure 75% marks (1.5 Marks) for the assigned task/question.

III. Student code shows more than 50% implementation of the given task. With few syntax errors and comments are not put up properly. In this case the student will be able to secure 50% marks (1 Mark) for the assigned task/question.

IV. Student code shows more than 20% implementation of the given task. With many syntax errors. Comments are not put up properly. In this case the student will be able to secure 0-25% marks (0 -0 . 5 Marks) for the assigned task/question.

Note: The above rubric is also utilized for different questions of higher-level score.

6. Conclusions

The tutorials and examination based on troubleshooting & debugging may help students to learn programming language in an interactive way. Also, IT industry seeks for candidate who is talented with enough knowledge of troubleshooting and debugging. This kind of pattern practice may help students in industry to work with/edit code written by peer programmers from same group/different groups. Learning as well as assessment based on such pattern will inspire the students to do much better for the programming course.

References

[1] Daly, C. (1999). RoboProf and an introductory computer programming course. ACM SIGCSE Bulletin, 31(3), 155-158.

[2] Nagappan, N., Williams, L., Ferzli, M., Wiebe, E., Yang, K., Miller, C., & Balik, S. (2003, February). Improving the CS1 experience with pair programming. In ACM SIGCSE Bulletin (Vol. 35, No. 1, pp. 359-362). ACM.

[3] Leal, J. P., & Silva, F. (2003). Mooshak: A Web- based multi-site programming contest system. Software: Practice and Experience, 33(6), 567-581.

[4] Zhenming, Y., Liang, Z., & Guohua, Z. (2003, November). A novel web-based online examination system for computer science education. In 33rd ASEE/IEEE Frontiers in Education Conference (pp. 5-8).

[5] Baylor, A. L., & Kim, Y. (2005). Simulating instructional roles through pedagogical agents. International Journal of Artificial Intelligence in Education, 15(2), 95-115.

[6] Ala-Mutka, K. M. (2005). A survey of automated assessment approaches for programming assignments. Computer science education, 15(2), 83-102.

[7] Smith, P. P. (2005). Techniques of Assessment Pertinent to Computer Programming Courses. In 38th Midwest Instruction and Computing Symposium, University of Wisconsin-Eau Claire, WI.

[8] Douce, C., Livingstone, D., Orwell, J., Grindle, S., & Cobb, J. (2005). A technical perspective on ASAP–Automated System for Assessment of Programming.

[9] Zhang, L., Zhuang, Y. T., Yuan, Z. M., & Zhan, G. H. (2006, July). A web-based examination and evaluation system for computer education. In Sixth IEEE International Conference on Advanced Learning Technologies (ICALT'06) (pp. 120-124). IEEE.

[10] Mandal, A. K., Mandal, C., & Reade, C. M. (2006). Architecture of an Automatic program evaluation system. CSIE Proceedings.

[11] De Raadt, M., Dekeyser, S., & Lee, T. Y. (2007). A system employing peer review and enhanced computer assisted assessment of querying skills. Informatics in education, 6(1), 163-178.

[12] Farrow, M., & King, P. J. (2008). Experiences with online programming examinations. IEEE Transactions on Education, 51(2), 251-255.

[13] Gárcia-Mateos, G., & Fernández-Alemán, J. L. (2009, July). A course on algorithms and data structures using on-line judging. In ACM SIGCSE Bulletin (Vol. 41, No. 3, pp. 45-49). ACM.

[14] Montoya-Dato, F. J., Fernández-Alemán, J. L., & García-Mateos, G. (2009, June). An experience on Ada programming using on-line judging. In International Conference on Reliable Software Technologies (pp. 75-89). Springer, Berlin, Heidelberg.

[15] Zhang, G., & Ke, H. (2010, January). SQL paperless examination system design. In 2010 Second International Conference on Computer Modeling and Simulation (Vol. 3, pp. 475-478). IEEE.

[16] Wang, Y., Li, H., Feng, Y., Jiang, Y., & Liu, Y. (2012). Assessment of programming language learning based on peer code review model: Implementation and experience report. Computers & Education, 59(2), 412-422.

[17] Rienovita, E. , & Takeuchi, Y. (2014).

Encouragement of skill mastering motive based on need for approval and Interaction. Human Interface, 235-238.

[18] Husain, M., Akki, M., Maralappanavar, M., & Narayan, D. (2015). Designing and Teaching of Language Independent Object Oriented Programming Course. Journal Of Engineering Education Transformations, 0, 248-251. doi:10.16920/jeet/2015/v0i0/59717

[19] Ranade, A. (2016). An Approach to Introductory Programming. Journal Of Engineering Education Transformations, 29(3), 26-32. doi:10.16920/jeet/2016/v29i3/85205

[20] Dontham, R., Patil, S. P., Patil, M. A., & Ingale, A. K. (2016). Educating IT-Engineering Theory Courses Using BLOOMS Taxonomy (TC-BT) as a Tool and Practical Courses Using Magic of Making Mistakes (PC-MoMM): An Innovative Pedagogy. Journal of Engineering Education Transformations, 29(3), 75-84.

[21] de Jesus Gomes, A., Santos, Á. N. F. S., das Dores Páris, C. P., & Martins, N. C. (2017). Playing with Programming: A Serious Game to Start Programming. In Gamification-Based E-Learning Strategies for Computer Programming Education (pp. 261-277). IGI Global.

[22] Ganesh, K. E., & Pranesha, T. S. (2018). Enhancement of Learning Outcomes through Implementation of best practices in Teaching Learning Process: A case study. Journal of Engineering Education Transformations, 32(1), 12-14.

[23] Lajis, A., Baharudin, S. A., Ab Kadir, D., Ralim, N. M., Nasir, H. M., & Aziz, N. A. (2018). A review of techniques in automatic programming assessment for practical skill test. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 10(2-5), 109-113.

[24] Sharma, A., & Arora, A. S. (2019). Active learning: an Instructional Technique for Improving Educational Practices. Journal of Engineering Education Transformations, 32(3), 36-39.

[25] Restrepo-Calle, F., Ramírez Echeverry, J. J., & González, F. A. (2019). Continuous assessment in a computer programming course supported by a software tool. Computer Applications in Engineering Education, 27(1), 80-89.

http://www.boss.org.uk/

http://placement.freshersworld.com/placement-papers