IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 27, 2019, accepted May 12, 2019, date of publication May 15, 2019, date of current version May 30, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2916979

A Rule-Based Method for Cognitive Competency Assessment in Computer Programming Using Bloom's Taxonomy

ZAHID ULLAH[1,2], ADIDAH LAJIS[1], MONA JAMJOOM[3], ABDULRAHMAN H. ALTALHI[2], JALAL SHAH[4], AND FARRUKH SALEEM[2]

[1]Malaysian Institute of Information Technology, Universiti Kuala Lumpur, Kuala Lumpur 50250, Malaysia
[2]Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia
[3]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh 11671, Saudi Arabia
[4]CSE&S Department, Balochistan University of Engineering & Technology, Khuzdar 89100, Pakistan

Corresponding author: Zahid Ullah (zasultan@kau.edu.sa)

ABSTRACT Assessment of students in computer programming is a challenge for instructors, especially at the introductory programming level, where the number of student enrollment is typically high. Therefore, this study presents a novel approach to assessing students' competency in programming using Bloom's taxonomy. The novelty of the presented approach is based on some rules that quantify the attained competencies with respect to the cognitive levels of Bloom's taxonomy. Unlike previous studies, in which cognitive levels were used as a scale for making the questions while the competency assessment was manually performed, in this study, the rule-based assessment method uses the automatic decision-making process to map the students' competency level directly to the corresponding cognitive levels from the written code without the prior mapping of questions to the cognitive levels. For this reason, the study focuses on the basic topics of the structured Java programming language (i.e. selection, repetition, and modular). The rule-based assessment method has been applied to students' programming code in the introductory level Java course. Data collection has been carried out through conducting an empirical test in which the valid responses of 213 students were collected, which was processed through the rule-based method for competency assessment. Moreover, the quantitative results achieved from the rule-based assessment method were validated by comparing them with the results achieved from the manual assessment. Furthermore, for comparative analysis, several statistical methods were used to identify the difference between the results of the two assessment methods. The outcomes of the comparative analysis have shown the reliability of the proposed rule-based assessment method.

INDEX TERMS Cognitive level, competency, assessment, Bloom's taxonomy, computer programming, rule-based.

I. INTRODUCTION

Computer programming is one of the more challenging tasks for computer science students, especially novice students in introductory programming courses [1]–[3]. Therefore, the number of students' failure in this course are typically higher [4], [5], or they produce unsatisfactory results every semester. In the literature, there are several reasons for students' difficulties in learning computer programming such as study methods, students aptitudes and attitudes, and psychological aspects [6], first experience of programming [1], [7], practicing new syntax and semantics [8]–[10], a complex integrated development environment, and large size classrooms [11]. However, it is unrealistic to expect a novice student to become a good programmer at the initial stage of his/her study because programming puts a heavy cognitive load on students [12], [13]. At the same time, students' competencies can be increased gradually by practicing more practical assignments as well as getting more feedback from the instructor.

Due to large classrooms, it is difficult for instructors to focus on the individual students [14], [15]. On the other hand, [16], stressed that assessment is a mandatory task for every educational institute because it determines student

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Asif.

competency and knowledge. However, according to [17], manual assessment of programming assignments is time-consuming, especially in large classrooms, and therefore needs extra human assessors to deal with increase in assessment workload. To address the issue of manual assessment of practical programming assignments, numerous automatic assessment tools have been developed by various researchers and academician [14]. These tools have been categorized into three analysis approaches: dynamic, static, and hybrid [8], [14]. In our previous research [14], we investigated and provided a comprehensive review of the strengths and limitations of the existing automatic assessment tools.

In this research, we have proposed a novel approach of a rule-based assessment method based on benchmark criteria that quantify students' competency level in programming and map it to the corresponding cognitive level of Bloom's taxonomy. Although there are several studies available [18]–[22] that have used Bloom's taxonomy for programming assessment, most of them used the taxonomy as a scale for making the assessment questions under each cognitive level and, more importantly, the assessment was performed manually. Moreover, there is no assessment method found that can assess the competency level automatically from the written code and map it to the corresponding cognitive level. To the contrary, the rule-based assessment method proposed in this research quantifies the students' competency level in programming and maps it to the corresponding cognitive level directly from the written code based on the requirements of meeting the criteria for that specific cognitive level. Furthermore, unlike the other studies, this approach needs no prior mapping of questions under the cognitive levels, but the cognitive levels will instead be predicted directly from the students' written code. This novel approach makes this research distinct from other studies. Moreover, the study has focused on the fundamental topics of programming: selection, repetition, and modular, which are not yet addressed in previous studies in the same domain of research. Furthermore, the case study selected for conducting this research was one of the leading higher education universities in Saudi Arabia. A total of 213 students' data has been collected and assessed through this rule-based assessment method. Similarly, the students programs were also manually assessed by the human instructors and compared the results. For comparative purposes, several statistical methods were used to identify the difference between the two assessment methods, as well as to determine the accuracy of the proposed assessment method.

The rest of the paper is organized as follow: section II discusses the competency assessment using Bloom's taxonomy, section III discusses methods used, results and results validation are described in section IV and V respectively. Section VI contains detailed discussion; section VII is the conclusion.

A. PURPOSE AND RESEARCH QUESTIONS

This study aims to present a novel rule-based approach to assess learning objectives, to quantify the students' competency in programming using Bloom's taxonomy, to enable automatic assessment for programing codes, and to answer the following research questions:

- How can Bloom's taxonomy be used to assess student competency levels?
- What is the assessment quality of the proposed assessment method?

II. BLOOM'S TAXONOMY AND COMPETENCY ASSESSMENT

Bloom and his colleagues [23] developed a taxonomy of educational objectives originally consisting of three domain: cognitive, effective, and psychomotor. Moreover, they determined the cognitive domain concerns of developing students' mental skills, the affective domain concerns of student attitudes, and psychomotor domain concerns of physical skills [23], [24]. Each domain has a taxonomy associated with it, in which all domains together form the goals of the learning process. In this study, we will address the student's cognitive skills.

In the cognitive domain, Bloom [23] developed the most widely used and cited taxonomy in education, which consists of a multi-tiered model known as Bloom's taxonomy [24]. The taxonomy provides a progressive platform of six cognitive levels of complexity, starting from lower (i.e., simple) to higher (i.e., complex) levels, which are: Knowledge, Comprehension, Application, Analysis, Synthesis, and Evaluation [23], [24]. The first three levels refer to lower complexity, while the later three levels are the high levels [24]. As stated by [25], mastery of each previous level is required to move on to the next level. Furthermore, a student performing well in the higher levels means that his/her performance in the lower levels must have been successful [26].

Bloom's taxonomy was considered a guide for assessing the learning objectives of a particular field, but after more close investigation, it has also provided a framework for the assessment of programmer knowledge [20]. Therefore, this taxonomy has been used in the computer science field [18] [27] for course design and assessment [19] and structuring assignments [26].

Several studies have used the cognitive levels of Bloom's taxonomy as a tool for the assessment of the students' competency in programming [16], [18]–[20], [22], [28], [29]. However, in most previous research, Bloom's taxonomy was used as a scale while the assessment process was manually performed. There was no study found that can automatically assess the students' competency directly from the student written code and map it to the corresponding cognitive level of Bloom's taxonomy. Furthermore, in these previous studies, the questions were prepared in advance for each cognitive level, while some of them such as [26], [30], [31], have suggested using MCQs as assessment questions for introductory courses. In this regard, the study of [32] classified MCQs according to the cognitive level of Bloom's taxonomy for assessing the students competency in programming in a

final exam. According to [32], they have used the scale for making questions as described by [33], i.e., easy, medium, and difficult. Furthermore, the authors have used the lower cognitive levels for assessment. Similarly, the study of [34] used Bloom's taxonomy as a scale for classifying the nine MCQs according to the revised Bloom's taxonomy. The study of [34] used three cognitive levels which are understand, apply, and analyze. Subsequently, the study of [29] proposed a rule-based approach for classifying assessment questions according to Bloom's taxonomy.

In summary, the previous studies have used Bloom's taxonomy as a tool for creating assessment questions, based on which students' competency levels was assessed. The approach used in this study is distinct in that the competency level will be assessed directly from the written programming code without prior mapping of the questions to cognitive levels.

III. METHODOLOGY

The current study focuses on cognitive competency assessment in programming using Bloom's taxonomy. To conduct this research, the study has proposed a novel assessment method for competency assessment based on standard criteria defined for each cognitive level in Bloom's taxonomy. The proposed assessment method has been applied to the students in a computer science department at one of the leading universities in Saudi Arabia. The step-by-step methodology of this study is discussed in the following section.

A. DATA COLLECTION AND SAMPLE

Since the competency assessment is based on the practical programming code, the data collection was based on the empirical test conducted with students in an introductory Java programming course. First, the assessment questions were formulated based on the opinions of three programming experts who had strong knowledge of the application of Bloom's taxonomy, in addition to their expertise in programming. The experts included a professor, an assistant professor, and a lecturer. The demographic data about the experts is shown in Table 1. The questions' formulation was completed in several sessions, liaising with experts. A total of three questions were finalized for assessment. The questions include: one simple, one medium, and one difficult in order to maintain justice among weak, medium and high competent students, which also support [32] and [33]. Moreover, the question formulation support previous studies in [6], [18]–[20], [29], [32], [34], [35].

Furthermore, per the suggestions described by [36] and [37], the translation of questions is sometimes mandatory, especially when English is not the first language in the region where data collection is being carried out. Thus, the assessment questions were translated into Arabic, since the study was conducted in Saudi Arabia. The translation was performed through one of the native Arabic speakers. In order to ensure the correctness of the translation, researchers retranslated the questions into English, and some minor corrections were performed to equalize both translations.

TABLE 1. List of experts selected for questions formulation.

Title	Role	Level of experience	Years of experience	Organization
Professor	Instructor/ coordinator	Senior	17	Education
Assistant professor	Instructor	Medium	04	Education
Lecturer	Lab instructor	Senior	13	Education

In the next step, an empirical test was conducted for the selected case study in which the valid responses of 213 students were received in the form of written codes. The exam was conducted at the end of Fall 2018 by a committee supervised by two of the experts, while one of the researchers was also present for support. The time allowed for the test was two hours and thirty minutes. Later on, 15 minutes of extra time was given, based on students' requests. The extra time given was supported by [32]. After completing the test, students were instructed to upload their answer files to Blackboard which were later on downloaded for competency assessment purposes.

B. PRE-PROCESSING

The students' programming files were downloaded from Blackboard, and the necessary steps were taken to prepare the files for assessment. First, the Java files were converted into text files because the proposed rule-based assessment method accepts the file as (.txt) format. The conversion of program files into text file was performed manually. The rule-based assessment method reads the code line by line; therefore, the code indentation was also performed through the NetBeans built function. Similarly, the comments if any provided by students in the code were cleaned to avoid interrupting the assessment process. The removal of the comments from the programs was done through our in-house built-in program, which cleans the code from comments automatically. Some other minor steps that did not affect the code but were helpful for processing through the proposed method were also performed.

C. CRITERIA-BASED BLOOM'S TAXONOMY LEVELS

This research aims to assess students' competency level in the basic topics of a programming course at the introductory level. Therefore, it is important to mention that each topic further consists of different structures, for example selection structures can be if(), if-else, or switch() in the program. The student's selection of these structures is solely at their discretion, for solving a problem shows the cognitive competency of a student. To track all these structures of the three main topics automatically, standard criteria needed to be met to produce the corresponding outcomes. Therefore, this study extracted numerous assessment criteria from the

literature, which are mainly based on the proposed work of [28]. Although the number of extracted criteria was much higher, a thorough review and investigations yielded only those criteria that were required for this study. Moreover, the selected criteria were further analyzed by experts who refined and finalized the criteria for each cognitive level. The refined assessment criteria for each cognitive level of Bloom's taxonomy are discussed in the following sections.

1) KNOWLEDGE

The first and lowest level of Bloom's taxonomy is Knowledge. At this level, students are required to prove that they remember either by recalling or remembering some concepts experienced in the education process [23].

In a programming context, at this level students recall the information from memory that are needed for writing a code and understanding the syntax of a structure used in the program [28]. For example, students at this level should be asked to name three structures of repetition or three methods used for I/O purposes [19], or to define a method in Java [29]. In short, students at this level are expected to recall from memory what they have learnt in the classroom [28]. Based on these statements, the following criteria were defined for each structure at each knowledge level.

Selection

1. Define the syntax of an if-statement using Java code (*for example, "if(){}"*)
2. Define the syntax of an if-else statement using Java code (*for example, "if(){} else{}"*)
3. Define the syntax of a switch statement using Java code (*for example, "switch(){}"*)

Repetition

4. Define the syntax of the for-loop using Java code (*for example, "for(;;){}"*)
5. Define the syntax of the while-loop using Java code (*for example, "while(){}"*)
6. Define the syntax of the do-while-loop using Java code (*for example, "do{} while();"*)

Modular

7. Define a method using Java code (*for example,* modifiers return-type modular-name (){}) [38].
8. Define a method using Java code (*for example,* modifiers return-type modular-name (parameter list){}) [38].

A student who provides the correct syntax of the structure used in the program meets the knowledge level. At this level students are not require to provide details; however, the definition of each structure used in the program in the form of Java code must be provided.

2) COMPREHENSION

The second cognitive level in Bloom's hierarchy is comprehension, which is related to the interpretation of information [28], [32]. Moreover, [23] explained that students are required to interpret the concept of a problem in such a manner that precisely elucidates its use when explicitly asked to perform. In a programming context, this level is defined by [39], the conversion of instructions from plain English or pseudocode into a single Java statement, for example, writing a code that declares a variable and holds an integer value. The ability of a student to interpret the given algorithm [29]. Furthermore, students at the comprehension level are able to understand that how each structure related to the topic works and its conduct [28]. Moreover, as stated by [40], students at this level should be able to explain the structure of a method in a code.

Based on the previous literature, this research has defined the following assessment criteria for the comprehension level:

Selection

9. Explain the structure of an if-statement (*for example, "if(condition){body of if statement}"*)
10. Explain the structure of an if-else statement (*for example, "if(condition){body of if statement} else{body of else part}"*)
11. Explain the structure of a switch statement (*for example, "switch (expression) {cases: statements; break;}*. The break statement is optional [41].

Repetition

12. Explain the structure of a for-loop (*for example, "for(initialization; condition; increment/decrement) {body of for-loop}"*)
13. Explain the structure of a while-loop (*for example, "while (condition)"*)
14. Explain the structure of a do-while-loop (*for example, "do{body of loop} while (condition);"*)

Modular

15. Explain the structure of method (*for example, "void method_name (parameters)"*)
16. Explain the structure of method (*for example, "return_type.*[1]*method_name (parameters)"*).

As this level pertains to explanation of the material, students are thus required to explain each structure used in the program in the form of Java code. For example, the students must understand the purpose of each structure used in the program.

3) APPLICATION

Application is the third cognitive level, which is related to applying and executing the learned information [28]. Students at this level use previous knowledge and experience of the same situation for solving a problem [21] or applying past knowledge to a new situation [19]. In the programming context, this level is defined by researchers as, *"the process and algorithm or design pattern that is known to students and applied to an unknown problem"* [16], [18]. The criteria defined by [29] for the application level to update the given situation of a for-loop with a while-loop. Furthermore,

[1] Return type means a method that returns value, excluding void method

as explained by [28], it measures student ability to use the appropriate structure of the topic required to execute the formula for a given problem-solving situation.

Based on the literature, this research finalized the following assessment criteria at the application level:

Selection

17. The range used in the condition part satisfies the required output (*for example, "if(x<=10)"*)
18. The range used in the condition part satisfies the required output (*for example, "if(x<=10) else if(x>10)"*).
19. The required number of cases provided and the correct case is executed as per the given expression (*for example, "switch(x) execute the required case"*)

Repetition

20. The number of iterations applied in *for-loop* execute per the required output
21. The number of iterations applied in *while-loop* execute per the required output
22. The number of iterations applied in *do-while loop* are as per the required output, and *do* structure must be executed at least once before testing the condition part (*for example, do{body of do structure}while(condition);*).

Modular

23. The applied *void method()* produces the required output. The void method does not return a value, but it should be called as a statement ending with a semicolon [42].
24. The applied *return_type method()* produces the required output. The return statement must be executed in method call [42].

The application level is the most critical level of Bloom's taxonomy because students are required to apply knowledge and execute the program to produce the correct output. For accurate output, they must provide the required condition (in case of control structures) or return type (in case of modular). As discussed by [28], novice students in programming seldom pass this level because it is related to the use of information to produce the required output.

Moreover, it is difficult to differentiate between the criteria defined for the comprehension and application levels. However, as discussed by [23] and supported by [22], the comprehension level is related to thinking based on what is precisely given, whereas the application level is related to bringing concepts from other previous experiences. This evidence made it clear for the above criteria defined for the application level.

4) ANALYSIS

The fourth cognitive level in Bloom's hierarchy is the analysis level, which is defined by [23], breaking down of information into its constituent parts and finding the relationships among them in an organized way. In programming, the previous statement is also supported in other studies [16], [19], [29], [39]. According to [18] and supported by [29], the breakdown of the program into parts and organizing those parts should be done in such a way that consistently achieves the objectives of the program. Similarly, [28] have defined the criteria at the analysis level, stating that students can solve problems by using the nested structures of the respective topics.

Based on the literature as discussed above, the following assessment criteria have been defined for assessing the students' competency at the analysis level.

Selection

25. The problem is analyzed using nested structure, for example, a nested if() statement is used [28], or multiple selections with *if()* statements are used (*for example, "if(condition1) {body of first if-statement} if(condition2){body of if the second statement}"* and so on.
26. Multiple selections of if-else() statement is used (*for example, "if (condition1) {body of if statement} then else-if(condition2){body of second if statement} then else-if (condition3)"){body of third if statement}*and so on, at the end, *else{body of else part}*
27. Nested selections of *switch()* statement are used (*for example, "switch (expression1){case 1: statement; switch (expression2) {case 2: statement;}}.*

Repetition

28. Nested for-loops are used for repetition structure (*for example, for(;;){body of outer for structure for(;;){body of inner for structure}}*or*for(;;){body of outer for structure while(condition){body of inner while structure}}*) and so on.
29. Nested while-loops are used for repetitions (*for example, while(condition){body of outer while structure,* then *while(condition) {body of inner while structure}},* or*while(condition){body of outer while structure* then *for(;;) {body of inner for structure}}*)
30. Nested do-while loops are used for repetitions (*for example, do {body of outer do structure,* then *do{body of inner do structure},*then*while (condition);*for inner do, and then *while(condition)* for outer do.

Modular

31. Multiple *methods()* are called for different tasks in problem solution.

At this level, students are required to solve the problem by breaking down the program into parts. This study has followed the criteria defined by [28] to use the nested structures of the corresponding topics for solving a problem. In this regard, the control structures can be used as nested, while the modular structures can be used by breaking the code into multiple methods and call from the main method to meet the definition of this level as defined by [23].

5) SYNTHESIS

Synthesis is the fifth cognitive level of Bloom's taxonomy and is related to either writing a completely new program or enhancing an existing one by making changes. According to the literature, students at this level are required

to create something new based on previously learned information [19], [21], [23], [29], [39]. Moreover, [28] stated that students at this level extract ideas from a variety of sources and synthesize the information before reaching a conclusion. Furthermore, [19] argued that students create a new program based on previously mastered ideas of selection, repetition, and modular.

It was also difficult to differentiate the criteria for the synthesis and application levels because they are closely related to each other. However, the statement of [18] offered a clear way to differentiate between the two levels. According to them, in application the students have experienced similar or nearly similar situations but applied on different data, whereas in synthesis, the students haven't experienced the same situation or algorithm, but they might have seen background information rather than the complete whole [18]. Moreover, [28] explained that at the synthesis level, students can decide which structure of the topic is appropriate for solving a problem.

Based on the above evidence, the following criteria have been defined for the assessment 0f students' competency level in programming at the synthesis level:

Selection

32. Construct code by applying the correct selection structure to fulfill the requirements (*for example, using if-else-if(), because the scenario cannot be solved with switch() selection*). For example, if the values to be calculated are float or double, then a switch statement doesn't work because switch statements do not accept float and double data types.

Repetition

33. Construct code by applying the correct repetition structure to fulfill the requirements (*for example, using while-loop because the scenario cannot be solved with the other 2 repetition structures or it will unnecessarily complicate the code while using the alternate structures*)

Modular

34. Construct code by applying the correct modular structure to fulfill the requirements (*for example, using a method that returns value because the void method cannot fulfill the requirements in those cases where the output is to be displayed in the main method)*

Students at this level are expected to synthesize and use the appropriate structure of each topic in the program. For example, in control structures, there is always one structure that produces the required output while the alternate structures are not appropriate since they might complicate the program. Therefore, the appropriate selection of the structures used is a sign of the cognitive ability of students at the synthesis level.

6) EVALUATION

Evaluation is the final and highest level of Bloom's hierarchy, defined by [23] as "*making a judgement about the idea, method, or material based on the criteria at which the particular are correct, effective, or satisfied*". In a programming context, this involves the ability of a student to evaluate the algorithm by making a judgment regarding the validity of an idea [19], [28]. Similarly, [18] argued for creating a program either from a new idea or from an existing idea that is new to a student. Further explained by [39], a student who designs a good program is assumed to be a good evaluator. Moreover, a student program that meets the standards of coding is assumed to be at the evaluation level [18], [29]. Furthermore, [28] concluded that a student who can defend the idea of using the switch() statement instead of other selection structures is at the evaluation level.

Based on the literature, this research finalized the following criteria for assessing students' competency in programming at the evaluation level:

Selection

35. Use *switch()* statement instead of *if-else* statement (for example, a situation where switch statement is more effective)

Repetition

36. Use the most effective repetition structure, for example, judge the scenario and choose the most effective structure that produces attractive outcomes without complicating the code. (For example, use a do-while loop if the statement is to be executed at least once).

Modular

37. Use *void method()*instead of a method that returns value (in cases where the question is required to display output with the same method), else otherwise.

Evaluation is the most complex level in Bloom's hierarchy because it is related to the judgment of a problem and the selection of the most effective structure to be used for problem-solving. It is obvious that one problem can be solved with different techniques, but there is always one technique that is more effective than others. Therefore, students at this level are required to judge the problem and select a structure related to the topic used that is more effective for problem-solving. For example, in selection structures, in some situations the use of a switch() statement is more effective than if() and if-else statements because the decision-making process is more robust for switch() statements. Similarly, in repetition structures, the use of a do..while() structure is sometimes more effective, especially when executing a statement or group of statements at least once. Similarly, in modular structures, the selection of an effective return-type method plays a vital role in making the program more effective and interesting.

Furthermore, the above 37 criteria defined for each cognitive level have been tested on the students' written code and quantified for the highest competency level a student achieves. The whole framework of the research design, including the criteria, is shown in Figure 1.

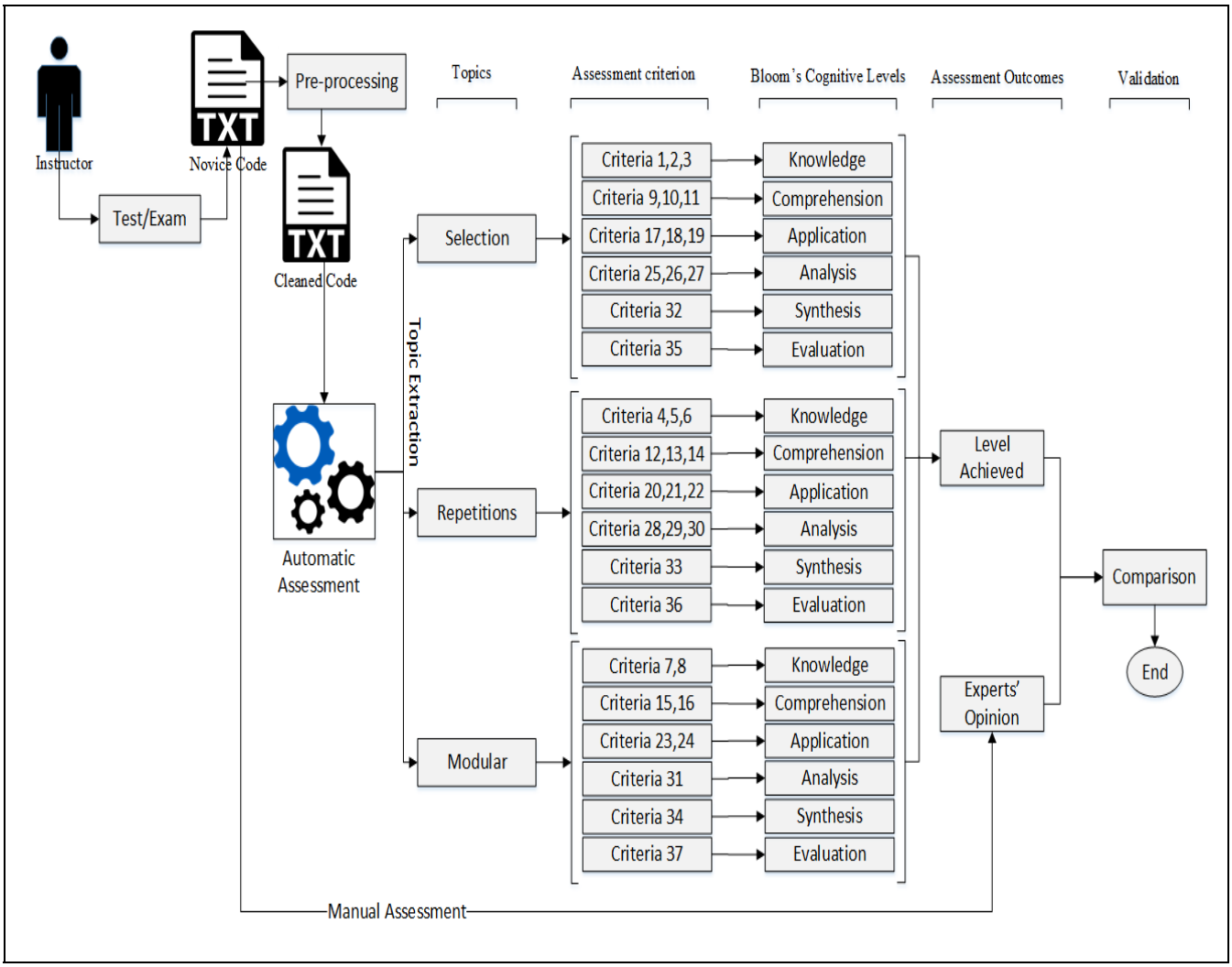

FIGURE 1. Competency assessment framework for basic topics based on bloom's taxonomy.

D. CRITERIA VALIDATION

In the first stage, researchers have extracted many criteria from the literature. The list of the extracted criteria has been discussed with experts, who suggested and finalized the 37 criteria discussed in the earlier sections, while unnecessary criteria have been discarded from the list. This was one advantage of this research in that the experts chosen to make the assessment questions and refine the criteria were active researchers and had a strong background in Bloom's taxonomy applications.

E. RULE-BASED ASSESSMENT METHOD

As discussed earlier, the proposed rule-based assessment method is purely based on standard criteria extracted from the literature that have been further refined and finalized by experts in the field. The selection of experts was based on their expertise in programming and, more importantly, their expertise in the application of Bloom's taxonomy as a tool for assessment. Therefore, this study has taken advantage of these experts and set the assessment criteria for each cognitive level in the algorithm of the rule-based assessment method. The criteria guided rule-based assessment method takes the students' programming files as input, while the competency assessment results after processing displays as output. For the competency assessment, a variety of formulas have been used in the algorithm to compute some measures in the form of software metrics for each input file. Some important metrics include total number of selection structures used, total number of repetition structures used, total number of modular structures used, the errors in these structure, and so many others, as shown in Figure 2. These metrics have played a vital role because they were used in a variety of equations that quantify the students' competency level with respect to their corresponding cognitive levels.

The procedure of competency assessment is simple in that the student programming file is input for the rule-based method, which tracks the entire code and searches for the topics used in the program. If any structure(s) of the basic

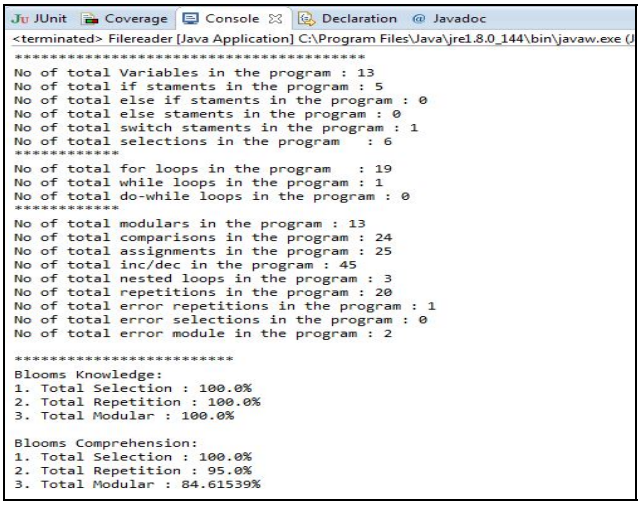

FIGURE 2. A sample output of a student performance and its mapping to the corresponding level of bloom's taxonomy.

topics are found in the code, they are evaluated against the assessment criteria, and the corresponding results are displayed as an output file. A student whose code satisfies the assessment criteria for the maximum cognitive levels has passed all the previous cognitive levels. A student who failed the first cognitive level will not be able to pass the next level because each preceding level is a prerequisite for the subsequent level. More interestingly, the use of a rule-based assessment method helps students in learning to programming in that for example, if a student fails a specific level, the instructor focuses on the criteria for that specific level instead of revising the whole code. This will definitely save time and effort for the instructor, especially in large classes.

Moreover, each topic has different types of structure, and in one program a student can use either different structures for the same topics or one structure multiple times, or even both. In this case more than one structure of the same topic are used; therefore, in the rule-based method the structures of each topic achieved have been calculated in percentages using equation 1.

$$Struct = \frac{TStructs - EStructs}{TStructs} \times 100 \quad (1)$$

where Struct calculates the percentage of specific structure for specific topic; TStructs is total number of structures used for specific topic; EStructs is number of errors occurring in each structure for a specific topic.

Furthermore, a sample output generated by the proposed rule-based assessment method is shown in Figure 2.

IV. RESULTS

The assessment results achieved with the rule-based method are discussed in this section. The results have been analyzed using SPSS version 24. Moreover, the descriptive statistics of the achieved results are exhibited in Table 2.

TABLE 2. Descriptive statistics (N = 213).

Cognitive Level	Basic topics	Min. Value	Max. Value	Mean	Std. Deviation
Knowledge	K_Selection	0	100	92.02	26.16
	K_Repetition	0	100	90.20	30.20
	K_Modular	50	100	96.24	12.77
Comprehension	C_Selection	0	100	86.15	32.54
	C_Repetition	0	100	83.80	35.30
	C_Modular	0	100	90.38	24.44
Application	AP_Selection	0	100	51.56	46.71
	AP_Repetition	0	100	46.25	48.60
	AP_Modular	0	100	55.40	46.30
Analysis	AN_Selection	0	100	21.99	38.80
	AN_Repetition	0	100	14.79	34.58
	AN_Modular	0	100	21.01	39.33
Synthesis	SY_Selection	0	100	15.05	33.78
	SY_Repetition	0	100	10.80	30.58
	SY_Modular	0	100	12.04	31.29
Evaluation	EV_Selection	0	100	6.32	23.48
	EV_Repetition	0	100	3.45	17.03
	EV_Modular	0	100	6.10	23.88

According to the results achieved from rule-based assessment method in Table 2, the mean values for all three topics in the first two cognitive levels are close to the maximum value, indicating that the students' competency levels are high. However, the graph has been dropped down at the application for which almost half of the total sample failed to achieve this level. The mean values for all three topics at the application level fall almost in the middle between minimum and maximum values, indicating that only half of the sample achieved the level. As the cognitive level progresses, the students' competency level is consistently regressing. At the analysis level, the students' competency level dropped further, and less than a quarter of the total sample achieved this level. Similarly, at the higher levels of synthesis and evaluation, competency in all three topics dropped to close to the minimum value, indicating that only a few students achieved the higher levels. Figure 3 shows the percentage of students achieving the corresponding cognitive levels in basic topics.

Furthermore, the students' cognitive competencies at the lower level were higher because the criteria for assessment were simpler. However, as the level increased, the competency level decreased because the assessment criteria were more complex. However, the results shown in Table 2 are not surprising because introductory-level students typically produce the same results. According to [28], the introductory-level students hardly reach the third level of Bloom's taxonomy. Similarly, [19] stated that a very small percentage of students can solve problems at the higher cognitive levels. Some other studies in the literature have suggested that introductory-level students should be assessed with respect to the lower levels of Bloom's taxonomy because the introductory course cannot produce accomplished programmers [30], which is supported by [22].

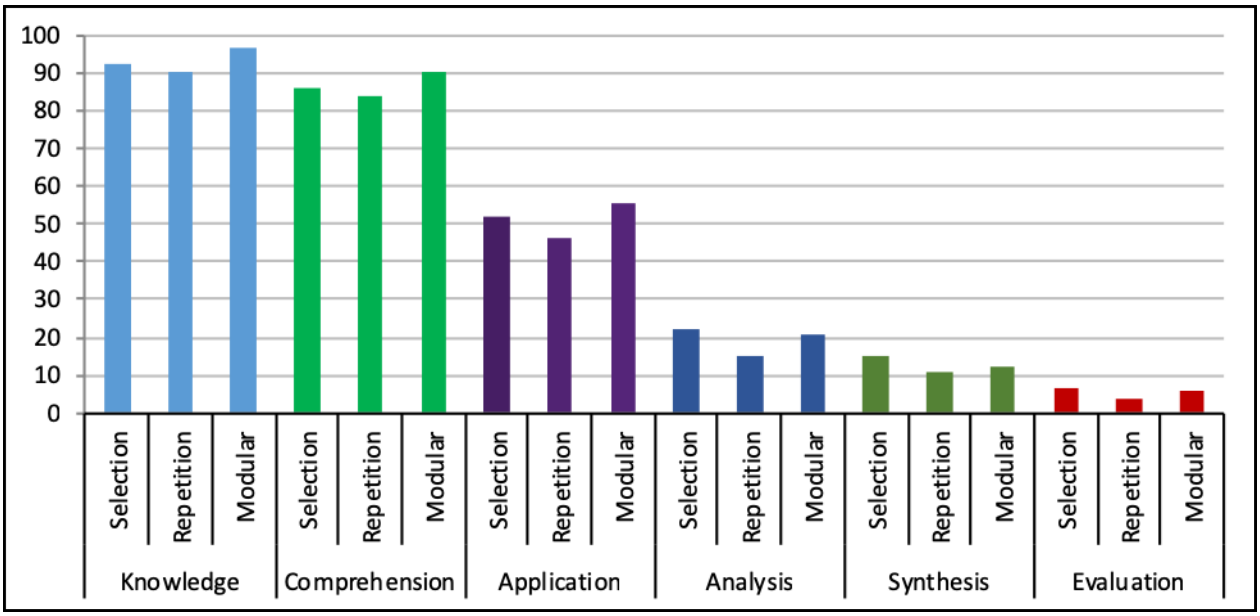

FIGURE 3. Percentage of students' competency in cognitive levels of Bloom's taxonomy.

It is obvious that students' performances at higher levels are disappointing. However, the aim of this study is to provide an automatic assessment method that can assess the students' competencies in programming at all six cognitive levels of Bloom's taxonomy. The research also aims to facilitate students in learning to computer programming. In this regard, the students' weaknesses with respect to a particular topic should be focused exactly where they are weak instead of revising the entire code. For that reason, if a particular cognitive level was not achieved, instructors should focus on the certain criteria for that specific level, rather than reviewing the whole code. This will ultimately save the time and effort of the instructor for other important activities.

Furthermore, the results achieved with the rule-based assessment method were also validated by comparison with the results achieved by manual assessment. The comparative analysis between the two assessment methods is discussed in the following section.

V. RESULTS VALIDATION

The results achieved with the rule-based assessment method have been validated by comparing them with those from the manual assessment method. For comparing the results, the experts were again requested to assess student competencies manually. For that reason, a total of sixty students programming files were randomly selected that were distributed among the three experts. Similarly, the same sample of students programs was again assessed through the rule-based assessment method in order to reveal the difference between the two assessment methods based on the same sample size. The results achieved with both the manual assessment method and rule-based assessment method were compared. The results of both assessment methods were further analyzed using various statistical methods. SPSS was used as a tool for the statistical analysis. Moreover, the methods used for comparative analysis were based on the method used in [17].

First of all, the results of the two assessment methods were analyzed using a paired sample T-test, and the effect size (d) was estimated for each structure in all topics. The effect size, which is commonly known as Cohen's d, is defined as the difference between the means of the two groups divided by the standard deviation [43]. Moreover, Cohen (1988) suggested three values of d, 0.2, 0.5, and 0.8 for small, medium, and large effect size, respectively. The effect size (d) shows the actual difference between the two assessment methods, as shown in Table 3.

As can be seen in Table 3, the effect size ($d = 0$, $p = 1$) is far less than the small effect size of 0.2, which is statistically non-significant with a p-value far larger than 0.05, showing that there is no statistical difference between the results achieved with the rule-based assessment method and manual assessment method in the first two cognitive levels of Bloom's taxonomy.

However, there were some difference in the results at the application, analysis, synthesis, and evaluation levels. A closer look at the values of (d) shows that most topics in the respective cognitive levels are less than 0.2, showing little difference. However, some values of (d) are a bit larger than 0.2, but not very large, so the difference is still in the small range. However, the overall effect size is statistically non-significant. The little difference between the results achieved with rule-based assessment and manual assessment

TABLE 3. Overall statistical analysis for the two assessment methods (N = 60).

Cognitive Level	Basic topics	Min. Value	Max. Value	Rule-Based		Experts' Judgment		Effect Size (d)	P-value
				Mean	Std. Dev	Mean	Std. Dev		
Knowledge	K_Selection	0	100	92.50	20.221	92.50	20.221	0.000	1.000
	K_Repetition	0	100	90.83	25.197	90.83	25.197	0.000	1.000
	K_Modular	50	100	97.50	10.989	97.50	10.989	0.000	1.000
Comprehension	C_Selection	0	100	87.50	27.036	87.50	27.036	0.000	1.000
	C_Repetition	0	100	84.17	32.536	84.17	32.536	0.000	1.000
	C_Modular	0	100	92.50	22.218	92.50	22.218	0.000	1.000
Application	AP_Selection	0	100	52.67	46.281	57.40	44.454	0.157	0.228
	AP_Repetition	0	100	46.12	45.451	51.95	46.951	0.221	0.092
	AP_Modular	0	100	57.80	41.682	63.08	40.374	0.177	0.175
Analysis	AN_Selection	0	100	20.33	40.252	24.33	42.321	0.238	0.070
	AN_Repetition	0	100	15.00	36.008	20.00	40.338	0.227	0.083
	AN_Modular	0	100	23.33	41.647	26.67	44.595	0.164	0.209
Synthesis	SY_Selection	0	100	13.33	34.280	16.67	37.582	0.105	0.419
	SY_Repetition	0	100	8.33	27.872	13.33	34.280	0.227	0.083
	SY_Modular	0	100	15.55	36.027	20.00	40.338	0.153	0.241
Evaluation	EV_Selection	0	100	8.33	27.872	11.67	32.373	0.184	0.159
	EV_Repetition	0	100	5.00	21.978	6.67	25.155	0.074	0.568
	EV_Modular	0	100	10.00	30.253	15.00	36.008	0.227	0.083

shows the reliability of the proposed assessment method for the competency assessment in computer programming. The overall percentage difference between the two assessment methods is shown in Figure 4.

VI. DISCUSSION

The main purpose of this research is to assess students' competency level in computer programming in an introductory-level course. Bloom's taxonomy was used as a scale for measuring competency. The research focused on the basic topics of programming for which the students' competency level was assessed. To conduct this research, a rule-based assessment method was developed that was purely based on the assessment criteria defined for each cognitive level. The criteria were taken from the literature and refined and finalized according to the opinions of the programming experts. The proposed rule-based assessment has been guided through these refined criteria that quantified the students' competency levels to the corresponding cognitive levels in Bloom's taxonomy.

The results of the rule-based assessment method were analyzed to show the success of the proposed assessment method. Since there was no such study found in the same domain to compare the achieved results for validation, researchers attempted to assess part of the sample through the human assessors manually. For that reason, the programming experts were chosen to assess the students' competency in the basic topics of introductory programming. The experts chose sixty students' programming files, which were distributed among the three experts. The results of manual assessment method from the three experts were further reviewed through one of the experts for the remaining two just for confirmation of the results, while the privacy of the experts was kept confidential.

The researchers have assessed the same sixty students' files chosen for manual assessment through the rule-based assessment method. The results of both assessment methods

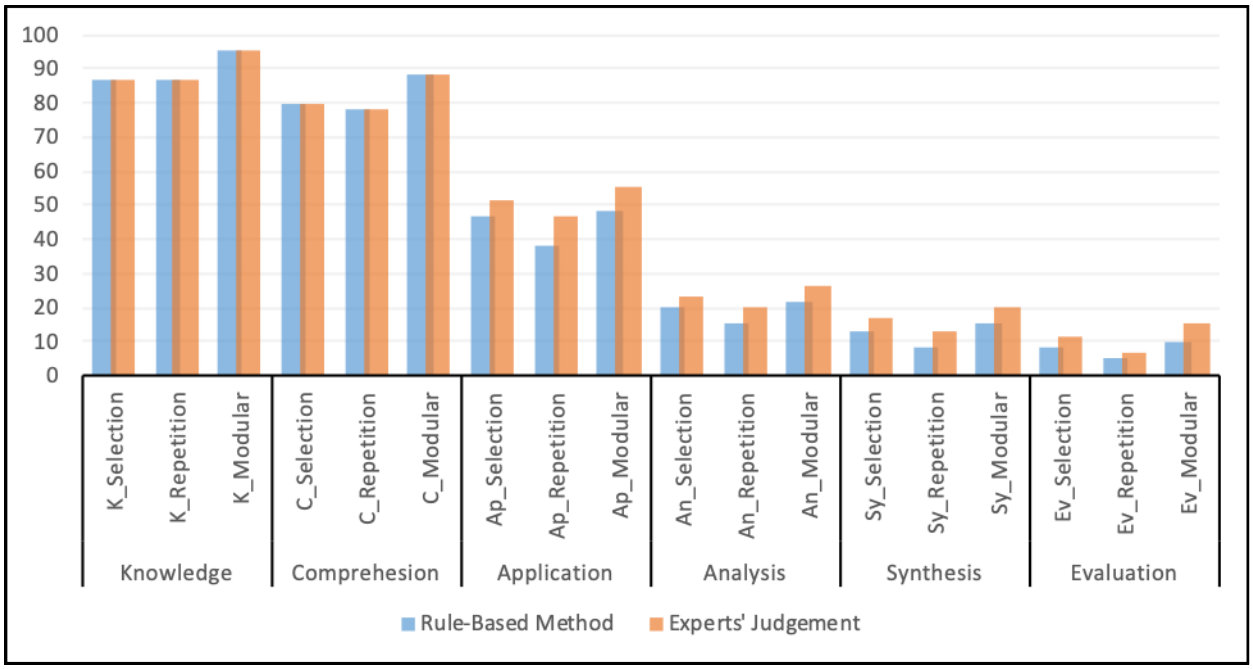

FIGURE 4. The overall percentage difference between manual and rule-based assessment methods.

were compared for validation. To further analyze the assessment results from both assessment methods, several statistical methods were used, using SPSS. The effect size (d) for each topic for each cognitive level was estimated, and a paired sample T-test was used. The results of the statistical methods exhibited in Table 3 revealed that the (d) values for most topics were lower than the small effect size, while some of the topics showed only a small difference. This indicates that there was little difference in the results when students were assessed manually and through the rule-based assessment method at the higher cognitive levels.

It is obvious that results will show a difference, especially when the human are involved and compared to the automatic process. In this regard, human intelligence plays a central role because it is flexible in assessment. For example, for a student code that is closely similar to the required one, the human assessors pass them with their intelligence to make decisions at the run time. However, automatic assessment methods strict adhere to the required criteria that a student must meet to achieve the goal. In this regard, the study of [17] also found a difference in results when they compared automatic assessment to manual assessment methods. According to them, automatic assessment has no similar distinctive power compare to the manual assessment method due to the lack of human intelligence in automatic methods. It has been further posited that for a program containing errors, manual assessment is more compassionate than automatic methods, and the automatic assessment method is proven to be more reasonable for highly competent students and less reliable for low-competency students [17]. Similarly, the same observation was made in the study of [44], in which differences between the two assessment methods were found. The above two studies mentioned had different objectives than those of this study. However, the methodology of the comparisons are the same, so this study used the same methods for comparison as the above two studies.

In contrary, the manual assessment can be influenced with the psychological status of the assessor as well as the assessment process for the same group of students in different time frames (instructor mood etc.). Consequently, the manual assessment is tedious for instructors especially in the large classrooms. Whilst, the rule-based assessment is more beneficial for the large classrooms and at the same time more advantageous for fair assessment because it needs to meet the required criteria irrespective of the time of assessment. Similarly, the assessment through rule-based is faster than manual assessment which can save the instructor time and efforts for other tasks.

VII. CONCLUSION AND FUTURE WORK

This study presented a novel assessment method for assessing the students' competency level in programming using Bloom's taxonomy. The competency assessment was carried out based on the standard criteria, which is a simple, easy, and convenient method for both instructors and practitioners since there are no rubrics, and generating test cases are not required. Moreover, the research questions in this study have been addressed with empirical evidence. The first research question was addressed in terms of using the cognitive levels of Bloom's taxonomy for the competency assessment in programming and the outcomes achieved successfully. Similarly, the second research question about the quality

of the proposed assessment was also addressed when the comparative analysis showed little difference from the manual assessment results. Moreover, the proposed rule-based assessment method can be used to help students enhance their competency level while learning programming. In this regard, a student can get help from the instructor in case a student fails to achieve a particular level, while the instructor can focus on the exact weakness of a student instead of reviewing the entire code. This ultimately enhances the students' learning programming as well as saving the effort and time of the instructor.

The results achieved with the proposed rule-based assessment method are statistically reliable and close to the manual assessment results. However, there are still some limitations worth mentioning. One of the limitations is that some assessment criteria are question-specific. Similarly, the proposed method was developed only for Java programming language.

In future, the few criteria that are question-specific should be generalized for the assessment of any programming code written in Java. Similarly, the same rule-based method can be used for assessing students in other languages such as C and C++, with some modifications of the criteria required for those programming languages.

REFERENCES

[1] K. Ala-Mutka, T. Uimonen, and H.-M. Jarvinen, "Supporting students in C++ programming courses with automatic program style assessment," *J. Inf. Technol. Educ., Res.*, vol. 3, no. 1, pp. 245–262, Jan. 2004.

[2] Y. T. Yu, C. K. Poon, and M. Choy, "Experiences with PASS: Developing and Using a Programming Assignment assessment System," in *Proc. 6th Int. Conf. Qual. Softw. (QSIC)*, Oct. 2006, pp. 360–365.

[3] S. Gul, M. Asif, W. Ahmad, and U. Ahmad, "Teaching programming: A mind map based methodology to improve learning outcomes," in *Proc. Int. Conf. Inf. Commun. Technol. (ICICT)*, Dec. 2017, pp. 209–213.

[4] M. A. Brito and F. de Sá-Soares, "Assessment frequency in introductory computer programming disciplines," *Comput. Human Behav.*, vol. 30, pp. 623–628, Jan. 2014.

[5] A. Robins, J. Rountree, and N. Rountree, "Learning and teaching programming: A review and discussion," *Comput. Sci. Educ.*, vol. 13, no. 2, pp. 137–172, Jun. 2003.

[6] A. Gomes and A. J. Mendes, "Bloom's taxonomy based approach to learn basic programming," in *Proc. World Conf. Duc. Media Technol. Assoc. Advancement Comput. Educ.*, Jun. 2009, pp. 2547–2554.

[7] Z. Xu, A. D. Ritzhaupt, F. Tian, and K. Umapathy, "Block-based versus text-based programming environments on novice student learning outcomes: A meta-analysis study," *Comput. Sci. Educ.*, pp. 1–28, Jan. 2019.

[8] D. Fonte, D. Da Cruz, A. L. Gançarski, and P. R. Henriques, "A flexible dynamic system for automatic grading of programming exercises," in *Proc.2nd Symp. Lang. Appl. Technol. SLATE*, vol. 29, 2013, pp. 129–144.

[9] G. Thorburn and G. Rowe, "PASS: An automated system for program assessment," *Comput. Educ.*, vol. 29, no. 4, pp. 195–206, Dec. 1997.

[10] J. C. Caiza and J. M. Del Álamo Ramiro, "Programming Assignments Automatic Grading: Review of Tools and Implementations," in *Proc. 7th Int. Technol. Educ. Dev. Conf.*, 2013, pp. 5691–5700.

[11] O. Aki, A. Güllü, and E. Kaplanoğlu, "An Application for Fundamental Computer Programming Learning," *Procedia—Soc. Behav. Sci.*, vol. 176, pp. 291–298, Feb. 2015.

[12] I. C. Mow, "Issues and difficulties in teaching novice computer programming," in *Proc. Innov. Techn. Instruct. Technol., E-Learning, E-Assessment, Educ.*. Springer, Dordrecht, 2008, pp. 199–204.

[13] D. Bau, J. Gray, C. Kelleher, J. Sheldon, and F. Turbak, "Learnable programming: Blocks and beyond," *Commun. ACM*, vol. 60, no. 6, pp. 72–80, Jun. 2017.

[14] Z. Ullah, A. Lajis, M. Jamjoom, A. Altalhi, A. Al-Ghamdi, and F. Saleem, "The effect of automatic assessment on novice programming: Strengths and limitations of existing systems," *Comput. Appl. Eng. Educ.*, vol. 26, no. 6, pp. 2328–2341, Nov. 2018.

[15] S. Buyrukoglu, F. Batmaz, and R. Lock, "Increasing the similarity of programming code structures to accelerate the marking process in a new semi-automated assessment approach," in *Proc. 11th Int. Conf. Comput. Sci. Educ. (ICCSE)*, Aug. 2016, pp. 371–376.

[16] A. Lajis, S. A. Baharudin, D. A. Kadir, N. M. Ralim, and H. M. Nasir, "A review of techniques in automatic programming assessment for practical skill test," *J. Telecommun. Electron. Comput. Eng.*, vol. 10, pp. 109–113, Jun. 2018.

[17] J. Liebenberg and V. Pieterse, "Investigating the feasibility of automatic assessment of programming tasks," *Inf. Technol. Educ. Innov. Pract.*, vol. 17, pp. 201–223, 2018.

[18] E. Thompson, A. Luxton-reilly, J. L. Whalley, M. Hu, and P. Robbins, "Bloom's taxonomy for CS assessment," in *Proc. 10th Conf. Australas. Comput. Edu.*, vol. 78, Jan. 2008, pp. 155–161.

[19] T. Scott, "Bloom's taxonomy applied to testing in computer science classes," *J. Comput. Sci. Coll.*, vol. 19, no. 1, pp. 267–274, Oct. 2003.

[20] J. Buckley and C. Exton, "Bloom's taxonomy: A framework for assessing programmers' knowledge of software systems," in *Proc. 11th IEEE Int. Workshop Program Comprehension*, May 2003, pp. 165–174.

[21] T. Kelly and J. Buckley, "A context-aware analysis scheme for bloom's taxonomy," in *Proc. 14th IEEE Int. Conf. Program Comprehension (ICPC)*, Jun. 2006, pp. 275–284.

[22] R. Gluga, J. Kay, R. Lister, S. Kleitman, and T. Lever, "Coming to terms with Bloom: An online tutorial for teachers of programming fundamentals," in *Proc. 14th Australas. Comput. Educ. Conf.*, vol. 123, Jan. 2012, pp. 147–156.

[23] B. S. Bloom, M. D. Englehard, E. J. Furst, W. H. Hill, and D. R. Krathwohl, *Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook I Cognitive Domain*. Harlow, U.K.: Longmans, Green Co LTD, 1956.

[24] M. Forehand, "Bloom's taxonomy from emerging perspectives on learning," Teach. Technol., Univ. Georgia, Athens, GA, USA, Tech Rep., 2010.

[25] L. W. Anderson, "Objectives, evaluation, and the importance of education," *Stud. Educ. Eval.*, vol. 31, nos. 2–3, pp. 102–113, 2005.

[26] R. Lister and J. Leaney, "Introductory programming, criterion-referencing, and bloom," *ACM SIGCSE Bull.*, vol. 35, no. 1, pp. 143–147, Feb. 2003.

[27] V. N. Pawar, P. B. Kudal, C. D. Patil, and P. S. Gaidhani, "Software Engineering Assessments using Blooms Taxonomy," *Int. J. Innov. Res. Sci. Eng. Technol.*, vol. 5, no. 11, pp. 19146–19152, Nov. 2016.

[28] A. Lajis, H. M. Nasir, and N. A. Aziz, "Proposed Assessment Framework Based on Bloom Taxonomy Cognitive Competency: Introduction to Programming," in *Proc. 7th Int. Conf. Softw. Comput. Appl.*, Feb. 2018, pp. 97–107.

[29] N. Omar *et al.*, "Automated analysis of exam questions according to bloom's taxonomy," *Procedia—Social Behav. Sci.*, vol. 59, pp. 297–303, Oct. 2012.

[30] R. Lister, "Objectives and objective assessment in CS1," *ACM SIGCSE Bull.*, vol. 33, no. 1, pp. 292–296, Feb. 2001.

[31] D. Traynor and J. P. Gibson, "Synthesis and analysis of automatic assessment methods in CS1: Generating intelligent MCQs," *ACM SIGCSE Bull.*, vol. 37, no. 1, pp. 495–499, Feb. 2019.

[32] S. Shuhidan, M. Hamilton, and D. D'Souza, "A taxonomic study of novice programming summative assessment," in *Proc. 11th Australas. Conf. Comput. Edu.*, vol. 95, Jan. 2009, pp. 147–156.

[33] F. M. Lord, "The relation of the reliability of multiple-choice tests to the distribution of item difficulties," *Psychometrika*, vol. 17, no. 2, pp. 181–194, Jun. 1952.

[34] J. Whalley *et al.*, "An australasian study of reading and comprehension skills in novice programmers, using the bloom and SOLO taxonomies," in *Proc. Conf. Res. Pract. Inf. Technol. Ser.*, vol. 52, Dec. 2006, pp. 243–252.

[35] D. Oliver and T. Dobele, "First Year Courses in IT: A Bloom Rating," *J. Inf. Technol. Educ.*, vol. 6, pp. 347–360, Jan. 2007.

[36] J. Harkness, B.-E. Pennell, and A. Schoua-Glusberg, *Survey Questionnaire Translation and Assessment* (Methods for testing and evaluating survey questionnaires). Hoboken, NJ, USA: Wiley, 2004.

[37] F. Karim, "Researching the acceptance and use of cloud computing for education and administration in saudi arabian universities, doctoral dissertation," Ph.D. dissertation, College Sci. Eng., Flinders Univ., Adelaide, SA, Australia, 2018.

[38] D. J. Eck, *Introduction to Programming Using Java*. 2006.
[39] R. Lister and J. Leaney, "First year programming: Let all the flowers bloom," *Proc. 5th Australas. Conf. Comput. Educ.*, Jan. 2003, pp. 221–230.
[40] D. A. Abduljabbar and N. Omar, "Exam questions classification based on Bloom's taxonomy cognitive level using classifiers combination," *J. Theor. Appl. Inf. Technol.*, vol. 78, no. 3, pp. 447–455, Aug. 2015.
[41] (2018). Tutorialspoint. *switch Statement in Java*. Accessed: Dec. 23, 2018. [Online]. Available: https://www.tutorialspoint.com/java/switch_statement_in_java.htm
[42] (2018). Tutorialspoint. *Java—Methods*. Accessed: Jan. 15, 2018. [Online]. Available: https://www.tutorialspoint.com/java/java_methods.htm
[43] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*, 2nd ed. 1988.
[44] Á. Matthíasdóttir and H. Arnalds, "Rethinking teaching and assessing in a programming course a case study," in *Proc. 16th Int. Conf. Comput. Syst. Technol.*, Jun. 2015, pp. 313–318.

ZAHID ULLAH received the master's degree in computer science from the University of Peshawar, Pakistan. He was a Researcher with King Saud University, Riyadh, Saudi Arabia, for five years. He is currently a Lecturer with the Information Systems Department, Faculty of Computing and Information Technology (FCIT), King Abdulaziz University, Jeddah, Saudi Arabia. His research interests include cognitive computing, business-IT alignment, IT business values, customer relationship management (CRM), and enterprise resource planning (ERP). He has published several papers in his research areas.

DR. ADIDAH LAJIS is currently an Established Lecturer and a Researcher in artificial intelligence and cognitive computing with the Malaysian Institute of Information Technology, Universiti Kuala Lumpur. Her research interests include natural language processing, text mining, cognitive assessment, and intelligent system. She actively participated in short terms research projects and received several research grants including Fundamental Research Grant Scheme from the Malaysia Ministry of Education. She is a reviewer of *Cognitive Computation* Journal and some of conference proceedings paper.

MONA JAMJOOM received the Ph.D. degree in computer science from King Saud University. She is currently an Assistant Professor with the Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia. Her research interests include artificial intelligence, cognitive computing, and machine learning. She has published several research articles in her research field.

ABDULRAHMAN H. ALTALHI received the Ph.D. degree in engineering and applied sciences (computer science) from the University of New Orleans, in 2004. He served as the Chairman of the IT Department for two years (2007–2008) and the Vice Dean of the College for five years (2008–2014). He is currently an Associate Professor of information technology with the Faculty of Computing and IT, King Abdulaziz University, where he is also serving as the Dean of the Faculty of Computing and IT. His research interests include wireless networks, software engineering, and computing education.

JALAL SHAH received the bachelor's degree in computer system engineering from the Balochistan University of Engineering & Technology, Khuzdar, Pakistan, and the master's and Ph.D. degrees in software engineering from the Universiti Teknologi Malaysia (UTM), in 2018. He is currently an Assistant Professor with Balochistan University of Engineering & Technology.

FARRUKH SALEEM received the Ph.D. degree from the Universiti Teknologi Malaysia. He is currently a Lecturer with the Information Systems Department, Faculty of Computing and Information Technology, King Abdulaziz University. He has overall 15 years of experience in the education field, mainly in teaching and different management work. He has published several journals and conference papers, mainly in ICT evaluation, data mining, ERP, and IT with business alignment.