International Journal on Advanced Science Engineering Information Technology

Vol.8 (2018) No. 4
ISSN: 2088-5334

Auto-marking System: A Support Tool for Learning of Programming

Marini Abu Bakar[#], Mohd Isrul Esa[*], Norleyza Jailani[#], Muriati Mukhtar[#], Rodziah Latih[#], Abdullah Mohd. Zin[#]

[#] *Center for Software Technology and Management, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia*
E-mail: {marini, njailani, muriati, rodziah.latih, amzftsm}@ukm.edu.my

[*]*Educational Planning and Research Division, Ministry of Education Malaysia, 62604 Putrajaya, Malaysia*
E-mail: [2]m.isrul@gmail.com

***Abstract*—Computer programming requires skills in designing algorithms, understanding syntax, writing programs, as well as the ability to correct errors in order to produce good programs. These skills can be developed through much practice on a continuous basis. The students' proficiency in programming is measured by the number of exercises that can be solved correctly within a specified period. From past observations, it is discovered that most students were able to solve the problems given during laboratory sessions. However, their performances did not carry over to laboratory tests. This situation points to the possibility that the students might not have performed adequate self-practice in preparing for laboratory tests. In a student-centered learning environment, fulfilling the notional learning hours is essential to ensure that students are prepared to take their subsequent classes. Based on a constructivist-learning framework, this article reports the development and evaluation of a prototype system to assist in the self-learning of programming. The online Auto-marking Programming Exercise System was developed based on the UVa Online Judge as a benchmark. The system can provide real-time feedback to students immediately after the students submit their programs. This instant feedback is an essential characteristic of the constructivist approach to learning. This will help students learn to programme in a useful way. The system is tested and evaluated for usability by selected users from among instructors and former students of computer programming course.**

***Keywords*— competition based learning; self-practice; notional hours; continuous learning; instant feedback.**

I. INTRODUCTION

Computer programming is one of the core courses in the STEM study field, which comprises Science, Technology, Engineering, and Mathematics. Programming requirements in this field have made programming a compulsory course in most courses offered at the higher education level especially for students in Computer Science and Information Technology. It is essential that students master the basic concepts of programming so that they can prepare for later studying of new languages and tools [1]. In the era of the 4th Industrial Revolution, programming is an essential element in the learning system. Developed countries such as the United States and Japan have begun to introduce programming from as early as in the primary. In the United State, there have been a string of developments designed to bring Computer Science education into every primary and secondary school [2]. Recently, the government of Japan adopted new policy strategies, including a plan to make computer programming compulsory at all public elementary schools from 2020 [3].

Programming is a skill acquired through continuous practice. Students should always practice mastering this skill. To assist students in honing their skills in programming, assignment, and exercises covering a variety of topics should be provided. The number of problem-solving assignments and programming exercises completed by the students can reflect the level of programming skills mastered by them [4].

Competition based learning approach or also known as competitive learning as a programming training medium has attracted students to the programming field. This is because programming competitions have a competitive edge to them, such as that presented in games, in which students are most interested in as compared to the conventional methods [5]. Through competition approach, students will be motivated to compete with their friends in order to achieve victory. ACM-ICPC, Aizu Online Judge, and PKU Judge Online are among the well-known programming competition websites which are participated by programmers all over the world. However, the questions in most programming competition websites are challenging and only suitable for skilled programmers. Students who are still in the beginner stage

need questions that are more appropriate to their level. Hence, a learning environment for programming which applies the competition approach for programming course students at the basic level is highly essential [6].

Programming promotes meaningful learning, which challenges the students' thinking. This is embedded through a constructive process, which requires students to build and develop knowledge by linking concepts that are newly acquired with the existing knowledge and concepts that they have. Computer programming can be a great way to improve computational thinking skills and build problem-solving skills [7].

To produce a skilled programmer is a long process and requires many problems solving experience. Students' skills and abilities in solving problems can be analyzed through three different phases. The first phase: the student's knowledge and experience are less organized and difficult to achieve. The second phase: knowledge becomes hierarchical and easy to achieve. The third phase involves the reorganization of knowledge and the ability to link the knowledge learned to build unique knowledge and skills. This is the same as the requirement for a student to master problem-solving skills in mathematics. Programming also requires students to perform continuous training to sharpen their minds and skills in programming. Training provides much experience to students. The extensive experience in successfully solving programming problems then makes it possible for a programmer to have resources and materials available to be used to solve other problems in the future.

At the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, the first programming course is required for all students. This course includes lectures, tutorials, and laboratory sessions. The learning outcomes of this course give a strong emphasis on aspects of program writing skills. Students are given training in program writing during lab sessions every week. Assessment of programming skills is done through three laboratory tests on topics taught. From observations in the last few sessions, most students were able to solve the problems given during laboratory sessions. However, their performances in the laboratory tests were somewhat less promising. This situation may be because the students copied each other during the laboratory sessions and did not do enough self-practice in preparation for the laboratory tests.

Many studies related to basic programming course have been conducted. Among them is an analytical study on the student's interest and attitude to identify problems in writing programs [8], as well as a study on students' perceptions towards the use of PC2 to check lab assignments [9]. Nevertheless, there is still a lack of research on the notional hours of students learning time outside the class. The notional hour's guideline Malaysian Institutions of Higher learning is that one credit is equivalent to 40 notional learning hours [10]. Thus in a student-centered learning environment, fulfilling the notional hours is essential to ensure students are prepared to take the next class [11]. For example, after going to lectures, students need to review the topic before following through with tutorials and lab sessions. Furthermore, after finishing the lab sessions, students will still have to try additional questions to improve their problem-solving skills. However, the amount of time spent by the students in revising and carrying out self-practice cannot be ascertained. Further studies are needed to investigate how to encourage and ensure that students do self-practice on programming outside the classroom. Self-practice is important to improve the students' skills in programming.

Based on research [8], outstanding students have their initiative to do additional exercises and to ask the lecturer or tutor if they encounter problems. On the other hand, other students mostly feel the training and the examples provided by the lecturers are adequate, and they have no effort to do additional exercises. One of the excuses given is the difficulty in understanding the errors obtained in the program. The difficulty of seeking assistance or asking questions further breaks their spirits to do self-practice.

The method of encouraging self-practice is to provide a competitive environment that provides instant feedback in order to increase the students' self-esteem so that they can improve themselves for the better.

II. Material and Method

Learning programming involves both theoretical and practical learning. Both of these elements are important and interconnected. Practical learning is necessary to apply the theories learned in lectures, whereas theoretical knowledge is needed during practical training. In order to develop a system that will aid in the self-learning of programming, it must be based on a specific learning framework. The learning framework implements the constructivism theory, which helps students build their knowledge and skills through the experience of solving various programming problems. This learning framework is also integrated with competitive learning models by way of providing a medium for students to compete in solving programming problems. Each of the components of this framework is explained below.

A. Student Centred Learning

Student-centered learning is a strategy in which students play an essential role and actively involve themselves in the learning process [12]. Lecturers become facilitators who guide students and allocate more time to carry out learning activities, whether in groups or individuals [13]. In this approach, students control the teaching process. Control here means that the students are the ones controlling the procedures, time and evaluation to meet their different needs.

Through this approach, students are trained to engage in learning sessions actively. The burden of communicating during learning sessions is given to the students. Techniques such as problem-solving that require critical and creative thinking; involving students in simulation and role-play methods, using self-study and co-operative learning can also be used as methods for this approach. Overall, the implementation of student-centered learning strategy encourages teachers to provide opportunities for students to learn independently.

B. Competition Based Learning

Competition in learning takes place when students compete with each other for the best grade in the classroom. Students will be encouraged to complete an assignment or

test, faster and better than their peers. However, as a rule for every game speaks, if there is a winner, there will be a loser. Likewise, in a competition based learning environment, there will be unsuccessful students who lose in competitions. This kind of thing can trigger situations whereby students will face disappointment and feel inferior when comparing themselves to their winning peers [14].

A competition based learning environment can make students feel more enthusiastic about winning, rather than learning. Besides that, a hostile and tensed situation can be easily created when there is a gap between the winning students and the losing students. In a competitive environment, some combination of personality dominance and individual level of competence will define the values of the process, inevitably marginalizing weaker and less skilled team members [15].

Competition in learning will possibly work when all students in the classroom understand and master the learning materials tested. Furthermore, if a student considers the competition as a fun game that does not merely focus on winning or losing, competition based learning will be a useful learning style in the classroom.

A competition which is structured and well-balanced can be a useful and harmless way of motivating students to do their best. Having competitions as part of competition based learning can also help students to face the real world, where they have to compete with other people for their job and so on. Competition based learning will also give students valuable moral experience and lessons to prepare themselves for work or business.

C. Online Program Assessment System

A programming competition is a competition, which requires participants to solve the problems given by producing a program. The ACM International Collegiate Programming Contest (ACM-ICPC) and the International Olympiad of Informatics (IOI) are among the organizations responsible for organizing regular competitions. ACM-ICPC is the premier global programming competition conducted by and for the world's universities. For nearly four decades, ICPC has grown to be a renowned competitive educational program that has raised aspirations and performance of generations of the world's problem solvers in computer sciences and engineering. On the other hand, UVa Online Judge, Aizu Online Judge, and PKU JudgeOnline host online programming competitions, which are participated by programmers from around the world.

Currently, most programming competitions use the online assessment concept. Online assessment is a system that users can send solutions for questions provided for automated system checks [16] [17] and gives instant feedback [18]. The solution will be tested with the test data set provided. Test results will result in keywords such as Accepted, Wrong Answer, Run Time Error, Output Limit Exceeded, Time Limit Exceeded, Memory Limit Exceeded, and Compile Error. Accepted means the received program uses the correct algorithm, the memory usage and time do not exceed the prescribed limit, and the program produces the output as per asked by the question. The wrong Answer means that the program produces an output that is not the same as the requested answer and others in which each keyword has a specific meaning.

D. Methodology

The use of an appropriate development model is essential to ensure the smooth running of the project and to ensure quality work. The methodology used in the development of this system is the waterfall model. This methodology is chosen because of its structured approach that helps developers to stay on the right track and away from problems. There are five phases in the waterfall model that are the needs analysis phase, the design phase, the development phase, the testing phase and the maintenance phase. System specification and system design is the outcome of the requirement analysis phase and design phase respectively. The system is developed based on the specifications, design, tested using black box testing strategies, and usability tests. The system is improved based on the results of the test.

III. Results and Discussion

This section will discuss the specification and design, implementation as well as testing of the Auto-marking Programming Exercise System (APEs).

A. Specification and Design of the Auto-marking Programming Exercise System

Because a student's confidence in programming can only be obtained by doing many self-exercises that provide instant feedback on a continuous basis, a support system is needed to help accomplish the process.

1) System Specification: Fig. 1 illustrates the use case diagram for the APEs. The actors in this system comprise of students, problem setters, lecturers and system administrators. All users need to be registered by the system administrator to get access to the system.

The proposed Auto-marking Programming Exercise System has extensive programming exercises and can respond immediately to students so that students can proceed to do the next exercise as soon as possible. Questions are of varying degrees of difficulty so that students can practice and learn according to their performance and rhythm. After login, the student can conduct the exercises, view the delivery list of exercise answers, view the questions in the exercises, send the answer to the exercises and view the scoreboard. Problem setters have only access to questions, such as adding and updating questions as well as test data.

Lecturers, on the other hand, can create exercises and classes, search for students, view questions, view answers, and view scoreboards. Lastly, the system administrator has access to all components of the system.

2) System Architecture Design: The system architecture for APEs consists of APEs web applications and automated assessment applications, as shown in Fig. 2. Functional specifications for APEs web applications have been explained in the previous sections

The automatic assessment application integrated into this APE system is the MOE Contest Environment. The program code sent by the user will be compiled based on the selected programming language and implemented by entering a test

data set. The output generated from the implementation of the program is then compared to the output test data to verify the program. The results of the assessment will be displayed on the scoreboard. Fig.3 illustrates this automated assessment process.

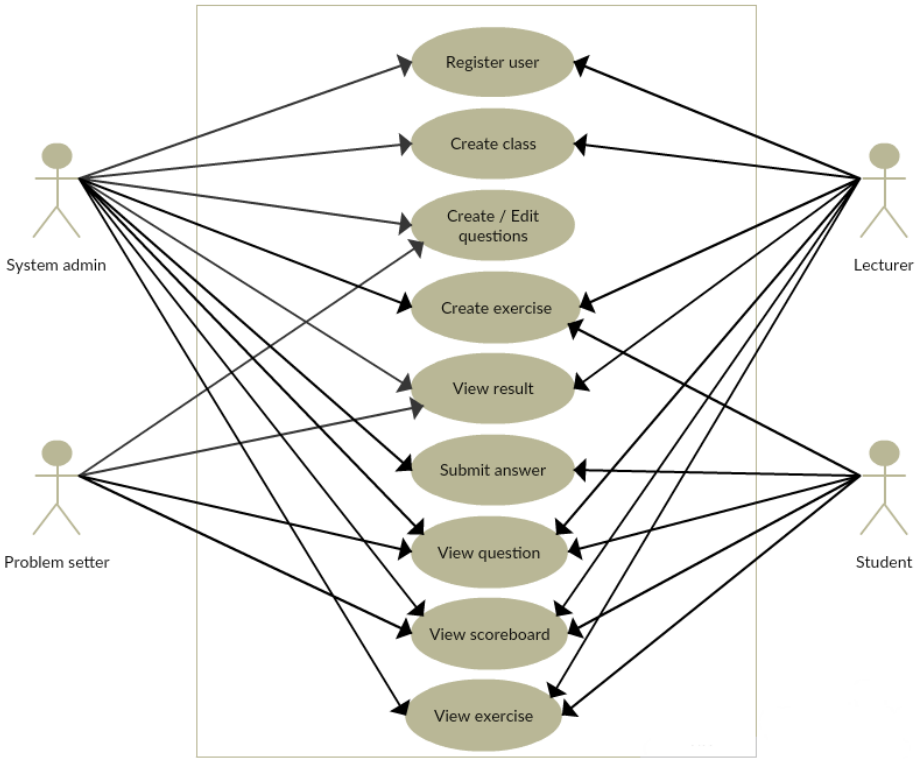

Fig. 1 The Auto-marking programming exercise system use case diagram.

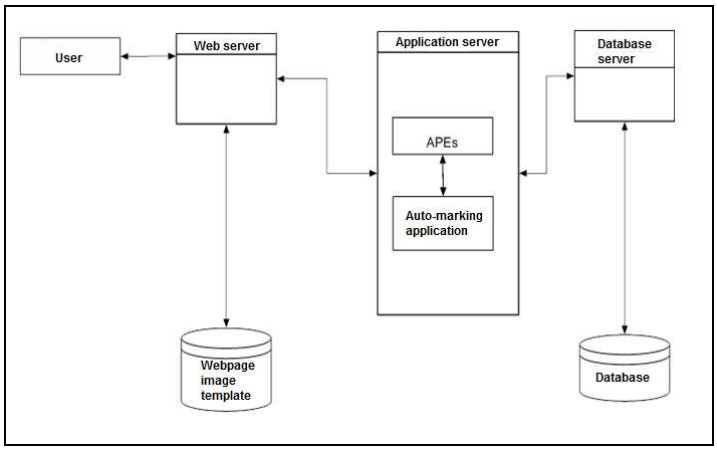

Fig. 2 APEs system architecture

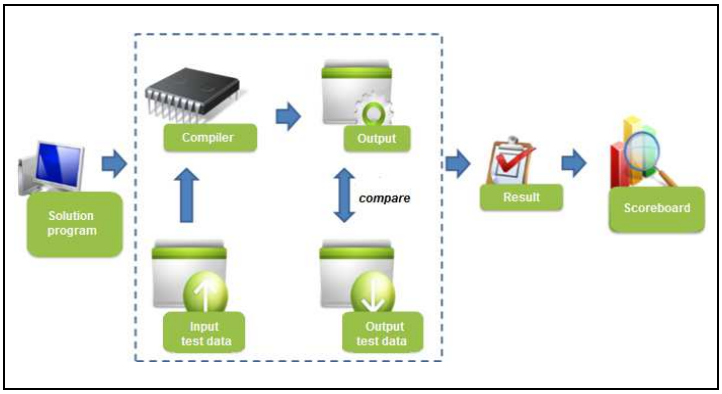

Fig. 3 Automated assessment process

B. *Implementation of the Auto-marking Programming Exercise System*

The Auto-marking Programming Exercise System consists of six modules. The modules are user management, question bank management, exercise management, class management, and scoreboard and automated program assessment. Each module is explained in the next section.

1) User Management Module: User management module function is to register users into the system. Only registered users can log into and have access to the system. There are four user categories namely system administrator, problem setter, instructors, and students. Users are categorized based on their roles. The only system administrator can register new users. Once a user logs into the system, the user dashboard will be displayed. APEs provides two types of user dashboards, one for students and a different type for the system administrator, instructors, and problem setter. The student dashboard displays a list of exercises assigned for the class the student is registered with (Fig. 4). The information displayed consists of (a) the exercise set title, (b) timestamp when the exercise starts and (c) end, (d) its status (ongoing or completed) and (e) a button for users to select the exercise. For ongoing exercises, students are allowed to submit answers while for completed exercises students are only allowed to review questions and answers.

The dashboard for the system administrator, instructor, and problem setter allows them to view APEs current operations. The dashboard displays a list of active exercises. The status for active exercises may be completed or ongoing. System administrator and other users can also monitor other users and graders (instructors) who are currently active. Status for auto-marking is also displayed to ensure it is ready to receive answer submission from students.

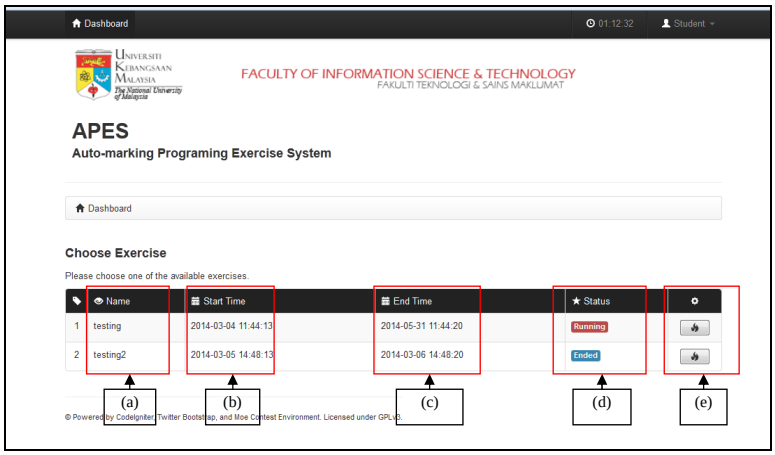

Fig. 4. Student dashboard

2) Question Bank Management Module: Programming questions are entered into the system via this module. Only users who are assigned the role of problem setter have access to the module. Details of the problem include problem name, author's name, time limit, memory limit, problem description, input format, output format, sample input and sample output as shown in Fig. 5. Problem setter will also have to prepare input and output test data sets to be used during program assessment in the auto-marking module. Other instructors while forming programming practice session according to topics can use questions in the question bank.

3) Class Management Module: Class Management Module is developed to assist instructors in monitoring and helping students in their programming practice sessions. This module allows instructors to form classes for a specific practice session. Instructors can select students registered by the administrator to join a specific class. Students practice programming and compete amongst each other in this class.

4) Exercise Management Module: Through the Exercise Management Module instructors can prepare programming exercises for students. Instructors can select suitable questions prepared by question setters according to a specific topic. This module requires instructors to set a time duration for students to solve programming problems. Only problems from the questions bank can be selected for exercise. If instructors want to create their problems, they have to log in as problem setters. When students click on a specific ongoing exercise of a particular topic, a list of problems will be displayed as depicted in Fig. 6(a). Next, students select a specific problem, and the question will be displayed in the ACM-ICPC format. Each problem has problem description, input, and output formats, as well as input and output samples. Button to submit solution (b) and remaining time for submission (c) are also displayed in this screen.

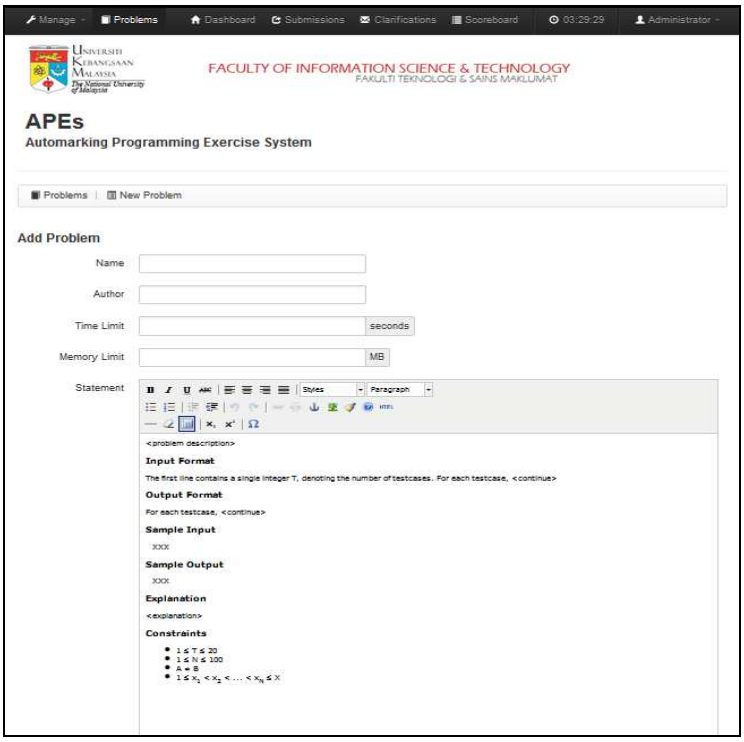

Fig. 5 Input problem interface

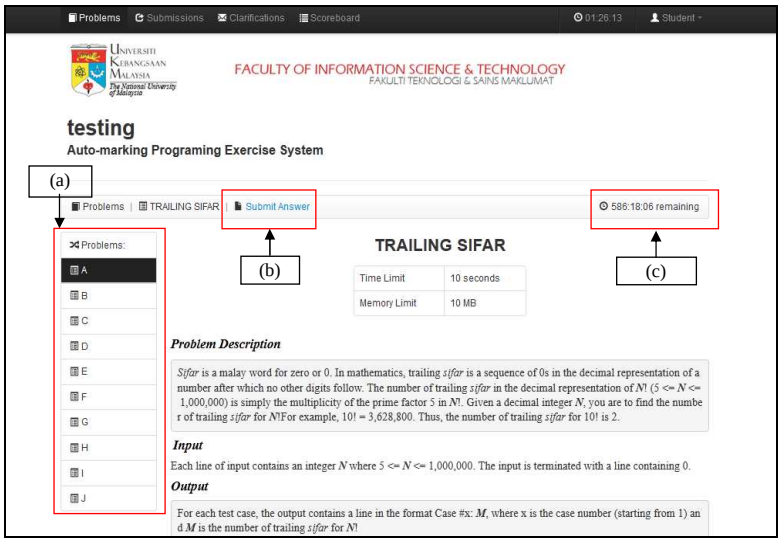

Fig. 6 Student exercise interface

To submit a solution, students will click on the submit button and a dialog box as shown in Fig. Seven will pop up. Next, students need to select a targeted (a) programming language, (b) upload the program solution file and (c) send the solution for auto-marking. When a student submitted a program, the list of all submissions will be displayed as depicted by Fig. 8. Information related to each submission include (a) the list of problem answered, (b) selected programming language, (c) timestamp of submission, (d) auto-marking verdict i.e. AC (Accepted), CE (Compile Error), WA (Wrong Answer), RTE (Runtime Error) and TLE (Time Limit Exceeded), and (e) icon to access details of auto-marking results.

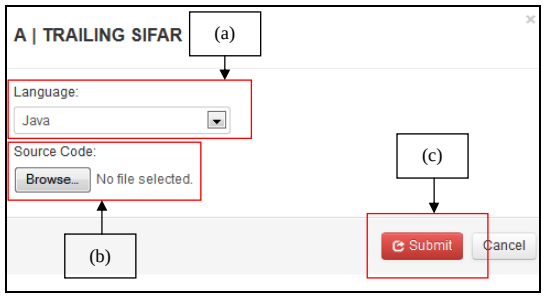

Fig. 7. Problem solution submission dialog box

5) Scoreboard Module: A score is given to every problem solved by students based on the duration and the number of attempts required to solve the problem. Students

are penalized 20 minutes for every failed attempt. The overall score is displayed on the scoreboard in ascending order. A student with the lowest score means that he or she is the fastest to solve the problem. Fig. 9 shows the scoreboard interface.

6) Automatic Assessment: This is the most important module in the Auto-marking Programming Exercise System. Each submitted program solution is compiled automatically. If the submitted program has a syntax error, the system will give error feedback, and the solution will not be accepted. If the submitted program is successfully compiled without error, the program will be evaluated with associated input test data. The generated output will be compared against the output test data. If they match perfectly, the submitted program is accepted and regarded as the correct solution, and the score collected will be displayed on the scoreboard. In contrast, if outputs compared are different, the system will give wrong answer feedback and score will not be recorded

C. System Testing

Testing for APEs is divided into two strategies namely the black box testing and usability testing. During the black box testing, each function is tested to ensure the modules developed to operate as stated and the outputs are by the system requirement. The black box test is divided into security testing, input data testing as well as out data testing.

Users to identify bugs, to test system effectiveness and to ensure that the system fulfils user's requirements, carry out usability testing. Individuals who require the individual users to use the system test the system interactively. The selection of respondents for the usability test is by randomly asking lecturers involved in the teaching of programming and former students who have taken programming courses. In total 11 respondents participated in this test.

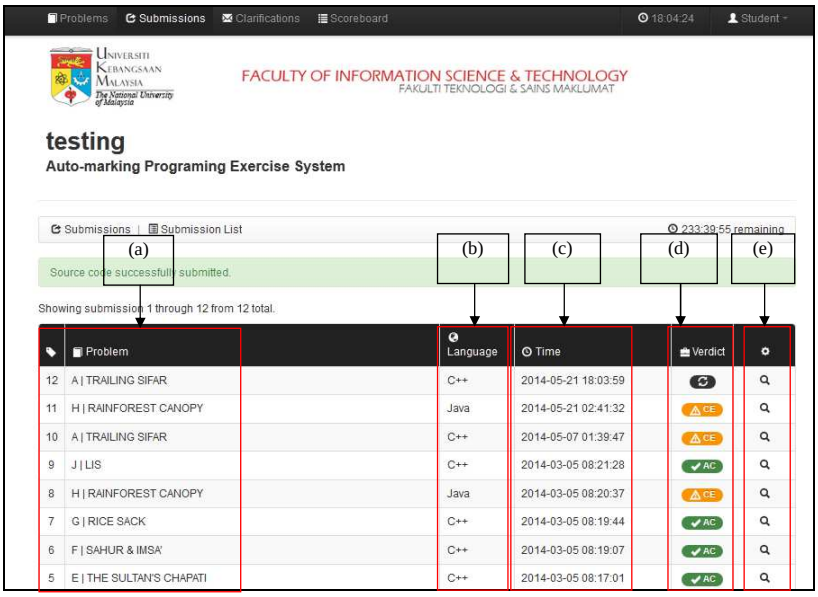

Fig. 8. List of submitted exercises

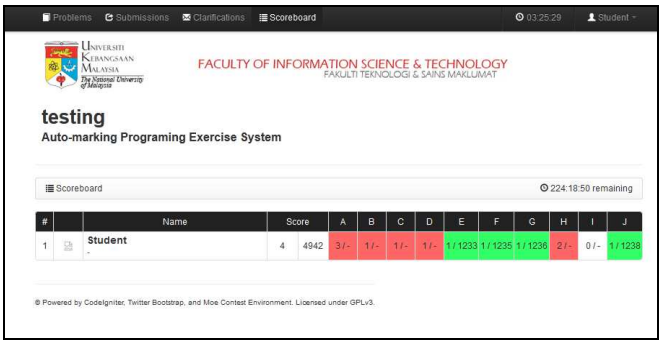

Fig. 9. Scoreboard interface

The primary aspects evaluated by users include:

1) System usability: System usability is evaluated based on whether it is easy to use fulfills the user's demand and requirement, and the user's readiness to recommend the system to students. Results from the test show that 58% of the respondents answered strongly agree, 33% answered agree, 9% answered not sure, and none of them answered disagrees or strongly disagrees.

2) System ability: System ability is evaluated based on its capability and effectiveness in performing all processes and modules in the system based on the models and approaches used. The result from the questionnaire shows that majority

of respondents agree that the APEs has the expected characteristics. The details result shows 24% of the respondents strongly agree, 76% agree, and none has chosen not sure, disagree or strongly disagree.

3) System user interface: The system user interface evaluation is carried out based on the functionalities and supports provided by the system. Also, the aspects of user-friendliness and understanding are also evaluated. The result from the questionnaire shows that 51% strongly agree, 49% agree, and none of the respondents have chosen not sure, disagree or strongly disagree.

IV. Conclusions

The Auto-marking Programming Exercise System has been developed and tested to evaluate its functionalities and usability. The APEs can provide some support. Firstly, it creates a question bank of programming exercises complete with their associated input and output data sets. Secondly, it supports a new class, supports for creating a new exercise based on selected questions from the question bank. Thirdly, it supports for accepting and evaluating programs submitted by students for selected exercises. Lastly, it supports for calculating, recording and displaying students' scores on a class's scoreboard. However, it can be further improved.

The auto-marking feature for APEs can perform the same functionalities as top online programming contest systems such as UVa Online Judge, Aizu Online Judge, and PKU JudgeOnline. It is also equipped with a question bank that is designed in such a way so that it is appropriate for novice programmers. The system also provides class management and user management functions so that teachers can monitor student performance in their respective classes.

It is every instructor's dream to provide a platform where students can perform self-exercise to improve computer-programming skills. Through such systems, instructors can monitor and identify students' weaknesses in specific topics and devise plans and methods to help them understand and master the topics.

By calculating the strategy of competition-based learning in the system developed it is hoped that instructors can create a healthy competitive environment to motivate students to perform programming exercises. To avoid students who sit at the bottom of the scoreboard from having low self-esteem, losing interest and focus towards programming courses, the instructors must explain the purpose of using such system in the teaching and learning process. Such a system should not emphasize on the rating on the scoreboard but rather on the efforts put in by students in practicing developing solutions to programming problems to gain experience through solving as many problems as possible.

Acknowledgment

We would like to thank the Ministry of Higher Learning, Malaysia for the earlier ground work on the project through its research grant fund FRGS/1/2011/SG/UKM/02/5 and to Universiti Kebangsaan Malaysia for supporting this work through its research grant fund PTS-2013-105.

References

[1] V. Aleksić and M. Ivanović, "Introductory programming subject in European higher education," *Informatics Educ.*, vol. 15, no. 2, 2016.

[2] Mark Nelson, "Computer Science Education in the Age of CS for All" *HuffPost*, 2016. [Online]. Available: https://www.huffingtonpost.com/acm-the-association-for-computing-machinery/computer-science-educatio_1_b_9373808.html. [Accessed: 16-Oct-2017].

[3] S. Murai, "Computer programming is seen as key to Japan's place in 'fourth industrial revolution,'" *The Japan Times*, 2016. [Online]. Available: https://www.japantimes.co.jp/news/2016/06/10/business/tech/computer-programming-industry-seen-key-japans-place-fourth-industrial-revolution/ [Accessed: 15-Sep-2017].

[4] A. Lishinski, A. Yadav, R. Enbody, and J. Good, "The Influence of Problem Solving Abilities on Students' Performance on Different Assessment Tasks in CS1," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16*, 2016, pp. 329–334.

[5] Nishimura Tomoharu, K. Shinichiro, and T. Hiroyuki, "Monitoring System Of Student Situation In Introductory C Programming Exercise With A Contest Style," in *ITHET*, 2011.

[6] Kurata, Tominaga, Hayashi, and Yamasaki, "Contest Style Exercise with Execution Tests for Every Lesson in Introductory C Programming," in *International Conference on Information Technology Based Higher Education and Training (ITHET 2007)*, 2007.

[7] S. Grover, R. Pea, and S. Cooper, "Designing for deeper learning in a blended computer science course for middle school students," *Comput. Sci. Educ.*, vol. 25, no. 2, pp. 199–237, 2015.

[8] M. Rahmat, S. Shahrani, R. Latih, N. F. M. Yatim, N. F. A. Zainal, R. A. Rahman, "Major Problems in Basic Programming that Influence Student Performance," *Procedia - Soc. Behav. Sci.*, vol. 59, pp. 287–296, 2012.

[9] R. Latih, M. A. Bakar, N. Jailani, N. M. Ali, S. M. Salleh, and A. M. Zin, "PC2 to support instant feedback and good programming practice," in *2017 6th International Conference on Electrical Engineering and Informatics (ICEEI)*, 2017, pp. 1–5.

[10] Malaysian Qualifications Agency, "Malaysian Qualifications Framework 2nd Edition," 2018.

[11] Rodziah Latih, Norleyza Jailani, Marini Abu Bakar and Zarina Shukur, "Pendekatan Pembelajaran Berasaskan Masalah dalam Kursus Pengaturcaraan Komputer" in *Inovasi Pengajaran dan Pembelajaran dalam Teknologi Maklumat*, Pusat Pengajaran & Teknologi Pembelajaran, UKM, 2015, pp. 104–111.

[12] P. K. Sevella and Y. Lee, "Determining the barriers faced by novice programmers," *Int. J. Softw. Eng.*, vol. Vol.4, no. 1, pp. 10–22, 2013.

[13] V. Goodyear and D. Dudley, "'I'm a Facilitator of Learning!' Understanding What Teachers and Students Do Within Student-Centered Physical Education Models," *Quest*, vol. 67, no. 3, pp. 274–289, 2015.

[14] D. W. Johnson and R. T. Johnson, *Learning Together And Alone: Cooperative, Competitive And Individualistic Learning*. Boston: Allyn and Bacon, 1994.

[15] J. Shindler, *Transformative classroom management : positive strategies to engage all students and promote a psychology of success*. Jossey-Bass, 2010.

[16] M. A. Revilla, S. Manzoor, and R. Liu, "Competitive Learning in Informatics: The Uva Online Judge Experience," *Olympiads in Informatics*, vol. 2, pp. 131–148, 2008.

[17] J. Bishop, R. N. Horspool, T. Xie, N. Tillmann, and J. De Halleux, "Code Hunt: Experience with Coding Contests at Scale," in *Proceedings - International Conference on Software Engineering*, 2015, vol. 2, pp. 398–407.

[18] H. Keuning, J. Jeuring, and B. Heeren, "Towards a Systematic Review of Automated Feedback Generation for Programming Exercises," in *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '16*, 2016, pp. 41–46.