COLLEGE OF COMPUTING AND INFORMATION SCIENCES

A programming assignment autograder for moodle

By

CS18-02

DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

A Project Proposal Submitted to the School of Computing and Informatics Technology For the Study Leading to a Project Report in Partial Fulfillment of the Requirements for the Award of the Degree of Bachelor of Science in Computer Science Of Makerere University

Supervisor

Ernest Mwebaze

School of Computing and Informatics Technology, Makerere University

emwebaze@gmail.com, +256-772-121-272

October, 2017

GROUP MEMBERSHIP

SNo	Names	Registration Number	Signature
1	ABULE AUGUSTINE ARUMADRI	15/U/2633/PS	
2	AYIKO Jeremiah Sara	15/U/186	
3	OWOMUGISHA ISAAC	15/U/12351/PS	
3	SENDIKADDIWA MARVIN	15/U/1154	

Contents

1 Introduction

1.1 Background

1.2 Problem Statement

1.3 Objectives

1.3.1 Main Objective

The general intention of the project is to develop a Moodle plugin that will automatically grade programming assignments uploaded to the system by students, keep track of each students performance based on scores attained from previously uploaded assignments and provide relevant feedback both to the students and the lecturer. The system should also enable lecturers to give in-class assignments for real-time assessment.

1.3.2 Specific Objective

- To design a Moodle plugin that will automatically grade the submitted programming assignments
- To implement a system that will keep records of submissions and scores attained, categorized according to student registration numbers
- To design a feedback system to both the lecturer and student, on basis of records stored
- To design a system that will detect plagiarized source code among submitted assignments.

1.4 scope

Computer programs and source code exist in many programming languages and size (in terms of actual lines of code), with different computer resource needs to efficiently execute. In order to serve a broad variety, it is important to build a general platform that can accommodate as many languages as possible. The aim of this project is to build a proof of concept platform. Thus, we only consider the C and Java programming languages. However, in the future we want the system to be platform independent.

When designing a platform used for automatic grading purposes, accurate assessment is of utmost importance. Given this, automatically generated programming assignments must be tailored with the intention of keeping the auto grading process simple and stable, at the same time challenging and relevant to students in order to hone their programming skills at the same time pass that particular course unit. Another important aspect is user feedback. The system is designed to be a substitute of the lecturer when it comes to assessment. Therefore it must provide relevant feedback to the students regarding use of better or correct algorithms, and also to the lecturer on topics where students are finding challenges.

For purposes of testing the effectiveness of the working system, we will focus on programming course units taught in the Computer Science course only within Makerere University. Programming languages covered in this course are C and Java.

1.5 Significance

A delivered working system will be of great help to both students and lecturers who use it as part of their teaching tools. Many at times students do not get the personal attention from lecturers as they so do desire. This may be attributed to the large number of students offering a particular course unit, each most likely with their unique challenges they face. There also exists the general perception that university students are not supposed to be spoon-fed, but rather engage in rigorous research and extensive personal reading. Much as this encourages academic independence, many students need one on one academic guidance. Our auto grading system provides this guidance.

By having the ability to suggest better algorithms and language structures/functions to use, each student can be guided on better coding techniques and how to optimize their programs. In the current setup at CIT, this is very difficult to achieve through manual means by the lecturer.

By enabling the lecturer to give in-class assignments that can be automatically graded, students are encouraged to constantly practice programming skills in order to be prepared for these assignments. This has a direct positive impact on their academic performance in that particular course unit, at the same time improving their general programming skills even though not examinable.

2 Literature Review

In this section, we provide a summary the previous work that has been done on E-learning platforms, auto-graders and online judges.

Many of the leading universities of computer science education around the world incorporate auto grading in their qprogramming and algorithms courses. For example, in 2013, MIT researchers Singh et al [11] introduced a new method for automatically providing feedback for introductory programming problems. According to [12] due to fair and timely feedback results from online judge websites, online practice outperforms traditional programming practice.

2.1 Related work

As far as E-learning is concerned in Ugandan Universities, the used platforms are not that efficient in reducing instructors efforts i.e. the Makerere University E-learning Environment (MUELE) platform is used for only publishing and collecting assignments for students. An extra effort has to be done on manually grading the submitted assignments.

The MUELE platform is owned by Makerere University and strictly bound to Makerere University Students. Due to the Unfavorable factors constraining the platform, it will be impossible to extend the project to the platform.
Designing and developing our project from scratch has problems overweighing advantages. One of these is rendering the project futile due to the competitions of other dominant E-learning platforms. Another problem associated with this is the development time required, a lot of time will be needed to get the design, user accounts, privacy and other security issues right if we are to create a project from scratch. Thus, this leaves us on cooperating and improving the available open source projects/platforms.

One of the most dominant E-learning platform used worldwide, Moodle. Moodle is a free and open-source software learning management system written in PHP and distributed under the GNU General Public License. Developed on pedagogical principles, Moodle is used for blended learning, distance education, flipped classroom and other e-learning projects in schools, universities, workplaces and other sectors. ("Moodle", 2017). The Moodle platform is composed of several features such components that enable its dynamism and thus can be adopted for the project.

Plugins are a flexible tool set, allowing Moodle users to extend the features of the site. ("Moodle", 2017). Plugins for moodle can be thought of modules or widgets with specific functionality. Currently moodle is maintained by thousands of plugins developed to suite different teaching curriculum. Amongst the plugin developed sofar include:- URKUND plagiarism plugin for checking plagiarism in submitted files, reengagement for sending timely feedbacks to students reminding them of uncompleted course activities. Some of these plugins being open source just as the platform they run (moodle), they can be used as sub modules in our plugin. For example, in order to solve the plagiarism detection stage in our project, the plagiarism plugin can be used as a sub component.

The most popular application of online judges is in programming contests such as TopCoder, ACM/ICPC and Google Code Jam. The most prominent of these is the ACM/ICPC [1]. The ACM/ICPC consists of algorithmic and programming problems for contestants to solve. The students submissions are evaluated based on the output of their programs. A typical problem in this contest consists of the following parts:

- Problem description: This is a description of the problem to be solved.
- Input description: This gives the specification of the input file from which the users solution programs read data
- Output description: This specifies the format in which the program should produce results.
- Sample input and sample output: These provide examples to clarify the input and output specifications.

For each problem in ACM/ICPC, there are multiple sets of data to be processed. This is an important feature that serves two major purposes [12]: first, they are used to test that the program can work in all possible situations (including corner cases). Second, they are used to calculate the running time of a program (a good solution should be able to work on both small datasets and larger datasets). The use of multiple datasets therefore ensures that a students submission is evaluated comprehensively.
Test datasets of problems in the ACM/ICPC are either generated by hand for small-sized datasets or by programs written by jury members that generate datasets according to some predefined patterns or at random [3].
In [7], Dr. Antti Laaksonen describes how a competitive programming approach using an online judge to teach algorithm concepts has helped improve the problem solving skills of the students at the University of Helsinki. In the course, the TMC system [9] is used. TMC allows students to create their Java solutions in the NetBeans IDE and evaluates the solutions using JUnit tests. The tests are written by the course staff. In this paper, he describes how this model of teaching helps students understand the importance of developing efficient algorithms. He also points out some limitations of using online judges. He gives an example of the following problem from [5]: Describe an $O(n)$-time algorithm that, given a set S of distinct numbers and a positive integer kn, determines the k numbers in S that are closest to the median of S. He points out the difficulty of automatically determining whether a students submission of the above problem is really $O(n)$, since the same problem can be solved in $O(nlogn)$ time. This therefore means that auto-graders are not perfect, however they seem to be better than traditional means of evaluation as the following section shows.

2.2 Benefits of auto-grading and online practice oriented teaching of programming

Researchers Wang GP et al. conducted two sets of experiments [12] in order to evaluate the effectiveness and validity of using their online judge system called OJPOT as compared to traditional teaching methods. They set out to answer the following questions [12]:

- Does OJPOT work effectively to enhance students practical abilities compared to the traditional teaching idea?
- In which aspects can OJPOT improve the students practical abilities?
- Output description: This specifies the format in which the program should produce results.

In the first set of experiments, two classes were chosen: one class defined as the control class (CC) and another defined as the experimental class (EC). In the CC, the traditional teaching idea was applied; while in the EC, the OJPOT online judge was used [12]. Before the experiment, both classes were given a pre-test to evaluate their elementary knowledge in C programming. Then during the semester, both classes were taught by the same teacher with the same timetable. The following data were collected during the experiment: pre-test scores, online practice and statistical data, post-test scores and course project and scores. Below is a summary of the results of their experiment ([12]):

- Both classes had similar programming foundation before the experimen
- At the end of the semester, there were significant differences between the CC class and the EC class: the EC class has overall better results in the final tests and project.

They conclude their results by remarking that the EC class performed better because the online judge system made the students more enthusiastic about the material they were studying. It was also observed that the general programming abilities of the EC class were improved much better than those of the control class

3 Methodology

3.1 Auto-grader/on-line judge

3.1.1 Method of grading

3.1.2 Limiting resources

3.1.3 Tools available for the lecturer

3.2 Analytics and other evaluation metrics

3.3 Moodle

4 References

5 Appendices

5.1 Appendix A: Activities Gantt chart.

5.2 APPENDIX B: Financial Requirements