AUTOMATED COMPUTER PROGRAM EVALUATION AND PROJECTS – OUR EXPERIENCES

UGC CARE APPROVED

ABSTRACT

This paper provides a few approaches to automating computer programming and project submission tasks, that we have been following for the last six years and have found to be successful. The approaches include using CodeRunner with Learning Management System (LMS) integration for programming practice and evaluation, and Git (GitHub) for project submissions and automatic code evaluation. In this paper, we describe the details of how we set up the tools and customized those for computer science courses. Based on our experiences, we also provide a few insights on using these tools for effective learning.

Keywords: *Computer programming, autograding, online assessment.*

BAMA SRINIVASAN
Department of Information Science and Technology

MALA NEHRU
Department of Information Science and Technology

RANJANI PARTHASARATHI
Department of Information Science and Technology

SASWATHI MUKHERJEE
Department of Information Science and Technology

JEENA A THANKACHAN
Department of Information Science and Technology

I. Introduction

Consider a typical institution conducting a programming course (lab) for students. Here, the student is expected to have a lab notebook and prepare the assigned programming assignment each week in a well formatted structure. This structure includes an algorithm, related programming segments known as code, and the output after successful completion. Once the course instructor verifies the logic of the code, the student types it to practice programming in the lab. In the first instance, the required output may not be arrived at and the student has to patiently debug the logical and syntactical errors, which takes time. Some students readily give up and take the easiest route of writing down only the output or copying from others. Course instructors now have these tasks at hand: (i) setting up the required programming environment in labs; (ii) verifying the logic of the code that has been written by students in notebooks; (iii) helping students to debug the errors; and (iv) evaluating students' work by checking the execution manually. Assuming a class size of 60, it is close to impossible for the course instructor to do all these tasks in the given time.

We have addressed these shortcomings by incorporating autograding program environments, namely CodeRunner (CR) and GitHub Classroom (GHC). Both environments provide a single interface for multiple programming languages, such as C, C++, Java, and Python. Hence, a separate setup environment is not required for each of these programming labs. With CR or GHC, course instructors distribute programming assignments to students with appropriate instructions, all in virtual mode. Then students work independently based on the instructions, and the code is verified automatically. Here, there is no manual evaluation, so the course instructor does not need to go to individual students' places and check the output of his code. The detailed work of attempts, debug methods, and marks is available to both the course instructor and the student instantly. This way, the course instructor can spend time helping students understand the programming part and debug the code if necessary.

Students are also expected to do a team project in their course of study, where they write code in a phased manner and develop software. To encourage the collaborative environment, we use GHC for the submission of students' code. This method provides the course instructors with information about code submissions, frequency of these submissions with changes, and the individual contribution, all in online mode. This reduces the manual time of checking the students' progress in evaluating projects. An additional advantage is that the codes are available in a repository that is accessible to the team, even at a later point in time.

In this paper, we provide the details of our experiences on setting up and using CR and GHC to automate the tasks of program evaluation and project submission. The paper is organized as follows. Section II discusses a few methods that are currently available for auto grading. Section III describes the details of the installation and customization of CR. Section IV provides an outline of GHC assignment submissions and the automatic evaluation of programs. This section also includes the details of project submissions through GHC. Section V gives a few insights from our experiences, and Section VI summarizes the paper.

II. Review of Existing Methods

Assessment and feedback are critical aspects of the learning and teaching process [3]. They not only benefit students, but also help professors identify course flaws. In programming based courses where practical skills are more important than memorization, regular practice and its evaluation are significant. Since the 1960's, various Auto Assessment tools for automatic program evaluation in such courses have evolved to assess the code with execution (dynamic) and without execution (static) [15]. TRY is a dynamic functionality assessment tool that was implemented in 1980's [33]. Assyst, Ceilidh (also known as CourseMarker), HoGG, Online Judge, and BOSS evaluate the program's functionality by comparing its output or checking the return values [12], [13], [23], [30]. WebToTeach and ELP tools evaluate single statements [1], [36]. Scheme-robo and JEWL evaluate codes in Scheme and Java with graphical user interfaces, respectively [11], [12]. Automata, Dr. Scratch, EduPCR, an AA system for learning object-oriented programming, VPL, Grading Java Assignments, GradeIT, and AA for OpenGL are some of the Automatic Assessment systems that have been used in the past few years [2], [9], [20], [22], [31], [34], [39].

Spanish National University of Distance Education (UNED) introduced Blocks, an open-source IDE which is limited to a few programming languages like C, C++, and Fortran [6]. PASS is a web-based automated Programming Assignment aSsessment System, introduced and used by the Department of Computer Science, City University of Hong Kong in the year 2004; it is not open source software [4]. Drop Project (DP) is an Auto Assessment tool used since 2018 at Lusofona University with various functionalities like standard assignment format, individual and group submissions, multiple submissions and tests, execution time and coding style [5]. DP is an open source tool but is limited to testing programming languages like Java, Kotlin, Clojure, and Scala. CodeZinger is a proprietary tool for the educators, with costs involved for each service, and the educators need to depend on this firm for every need [8]. Function specific autograders like Nbgrader and jupyter autograde are restricted with jupyter notebook only, limiting them mostly to Python [24], [32].

CodeRunner and Virtual Programming Lab are free, open-source plugins for the Moodle learning platform [7]. CodeRunner, extended with OpenGL programs plugged into the Moodle learning platform, was effectively used at the University of Auckland, New Zealand, for a Computer Graphics course (200+ students).

Git is the most prevalent version control system (VCS) nowadays [16]; in the 2021 Stack Overflow survey, more than 93% of professional developers reported using Git. As a result, there has been an increase in the use of Git in university courses. Among educators, the hosting service GitHub (GH) and its programming education tool GitHub Classroom (GHC) are particularly popular. Git workflow, autograding, offline automated processing, pull requests, admin rights to students, GitHub organization, submission dates and times, linkage with LMS like Moodle or Canvas constitute a few among many professional features [17], [18].

From this study, it is evident that most of the tools are proprietary and therefore expensive, which does not fit the educational requirements for serving a large number of students. Some of the tools are program specific, which means they need separate resources and support for installation and customization. However, tools such as CR, GH, and the Virtual Programming Lab are free for the educational community. We have chosen CR owing to its easy customization and support. Since GH is prevalent and working with it is beneficial for students from an industry standpoint, we have incorporated GH in our courses and programming labs.

III. Automated Code Evaluation with CodeRunner

CodeRunner is an automated method for evaluating programming exercises. It was first developed and used by University of Canterbury [7] as a plugin for Moodle [25]. We installed this tool in our department labs and used it to run programming courses and assessments. In this section, we provide the details of installing, customizing, and using this tool for our courses.

A. Installation, Configuration and Customization

Two servers with the following configuration were assigned for the purpose of implementing Learning Management Systems: Intel Xeon E3-1200 v5, RAM of 32 GB, and Hard Disk of 16 TB. Both servers run Linux, one with CentOS and the other with Ubuntu. Moodle was installed in the former one and was used for all courses. Without disturbing this setup, we have installed the CodeRunner tool on the other server through the sandbox that supports small programming tasks, namely, Jobeserver [7]. The design of the setup is given in Figure 1. The steps of the installation of Moodle initially were carried out with Apache, MySQL, and PHP [27]. For the CR, the Jobeserver was installed first, along with Apache and the related programming packages such as Python 3, C, C++, Java, and Octave [19]. Since it is a private server, the API authentication mechanism was not installed. Once installed, this server was connected to the Moodle server through its IP address or domain address. For our setup, we have configured the domain address as "auistjobe.edu."

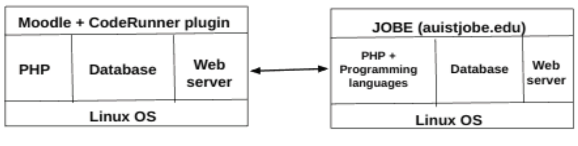
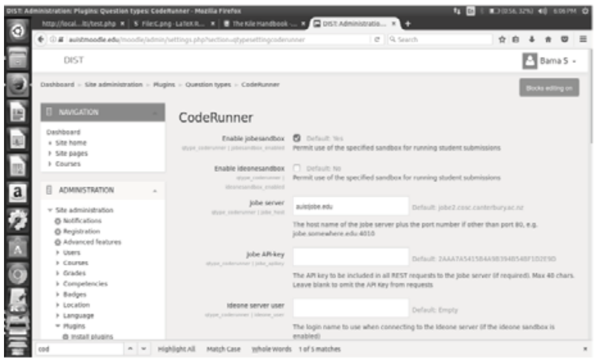


Figure 1 Setup of Moodle and Code Runner

By default, the address of the plugin is Canterbury web server, namely 'jobe2.cosc.Canterbury.ac.nz'. This however is not a good option because: (i) the number of students accessing CR from our department is more than 100 and (ii) the server has to be connected to the internet if default settings are used. These two limitations are addressed by having our own sandbox installed in a separate server as mentioned earlier.

From Moodle administration, first the plugin of CR was installed [7]. Then in the configuration option, the domain address of the machine with Jobeserver was given, which is shown in Figure 2.



CodeRunner

Enable jobesandbox [✓] Default: Yes
Permit use of the specified sandbox for running student submissions

Enable ideonesandbox [ ] Default: No
Permit use of the specified sandbox for running student submissions

Jobe server: auistjobe.edu — Default: jobe2.cosc.canterbury.ac.nz
The host name of the Jobe server plus the port number if other than port 80, e.g. jobe.somewhere.edu:4010

Jobe API-key — Default: 2AAA7A5415B4A9B394B54BF1D2E9D
The API key to be included in all REST requests to the Jobe server (if required). Max 40 chars. Leave blank to omit the API Key from requests

Ideone server user — Default: Empty
The login name to use when connecting to the Ideone server (if the ideone sandbox is enabled)

Figure 2 Customization through domain address

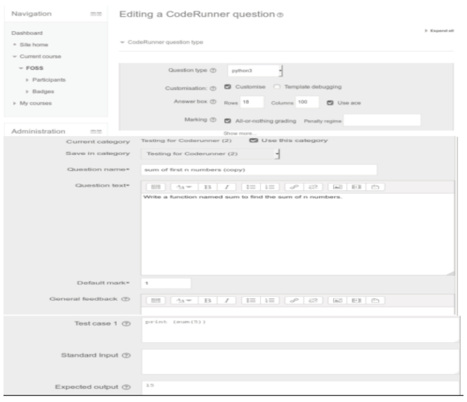
We have further customized the whole setup by having two domain addresses - one which can be operated only within the labs without internet connection and the other, which can be accessed anywhere in the department/ university with internet connectivity. This customization helps the teachers to load questions in Moodle from their place and students can answer those questions within the lab premises without internet connectivity as a closed book assessment.

B. Code Runner Usage

CR supports multiple languages such as C, C++, Java, Python, PHP and Octave. Once the plugin of CR is installed, the question bank options from Moodle has CR, through which programming questions can be added as an adaptive mode.

The teacher has the flexibility of providing the customized or debugging options with the required test cases. Figure 3 shows these options for a sample question of finding the sum of first n natural numbers for the programming language Python with a test case. Here, any number of test cases can be added with this setup.

With these questions in the question bank, the teacher can provide a quiz from the course activity option in Moodle as an adaptive mode. Students can view the question and code within the window and debug with the check option. Figures 4 and 5 show the attempts of students for correct and wrong answers, respectively.



Editing a CodeRunner question

Question type: python3
Customisation: [✓ Customise] Template debugging
Answer box: Rows 18 Columns 100 [✓ Use ace]
Marking: [✓ All-or-nothing grading] Penalty regime
Current category: Testing for Coderunner (2) [✓ Use this category]
Save in category: Testing for Coderunner (2)
Question name*: sum of first n numbers (copy)
Question text*: Write a function named sum to find the sum of n numbers.
Default mark*: 1
General feedback
Test case 1: print (sum(5))
Standard input
Expected output: 15

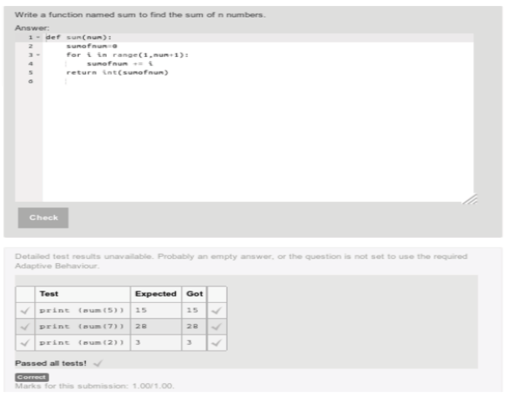
Figure 3 Code Runner - Adding Question

We have conducted programming assessments for multiple languages to more than 450 students simultaneously using this setup. Students can debug the code and work until they get the correct answer according to the test cases. Teachers also get the immediate feedback and marks of each student.

While CR is used within our University settings, we also explored GHC for online programming tasks. This is described in the next section.

IV. Use of GitHub for Programming Courses and Projects

Git is a free and open source distributed version control system used by software developers for development of software [16]. GitHub (GH) is an internet hosting service for software development based on Git [18]. It provides extensive support with free resources for teachers and students through GitHub Classroom (GHC) [17]. In the next section, we describe the details of setting up classrooms for our courses.



Write a function named sum to find the sum of n numbers.

Answer:

```
def sum(num):
    sumofnum=0
    for i in range(1,num+1):
        sumofnum += i
    return int(sumofnum)
```

Detailed test results unavailable. Probably an empty answer, or the question is not set to use the required Adaptive Behaviour.

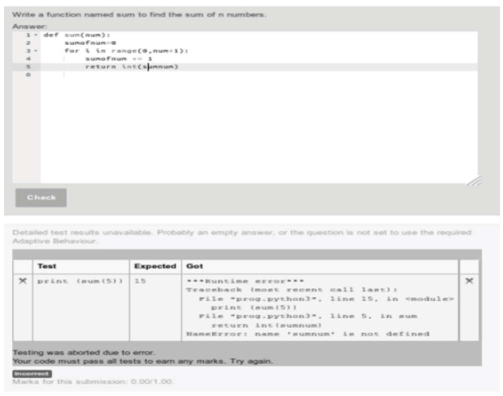
	Test	Expected	Got	
✓	print (sum(5))	15	15	✓
✓	print (sum(7))	28	28	✓
✓	print (sum(2))	3	3	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Figure 4 Correct answer



Write a function named sum to find the sum of n numbers.

Answer:

```
def sum(num):
    sumofnum=0
    for i in range(0,num+1):
        sumofnum += 1
        return int(sumnum)
```

Detailed test results unavailable. Probably an empty answer, or the question is not set to use the required Adaptive Behaviour.

	Test	Expected	Got	

✗	print (sum(5))	15	***Runtime error***
			Traceback (most recent call last):
			  File "prog.python3", line 15, in <module>
			    print (sum(5))
			  File "prog.python3", line 5, in sum
			    return int(sumnum)
			NameError: name 'sumnum' is not defined
				✗

Testing was aborted due to error.
Your code must pass all tests to earn any marks. Try again.

Incorrect

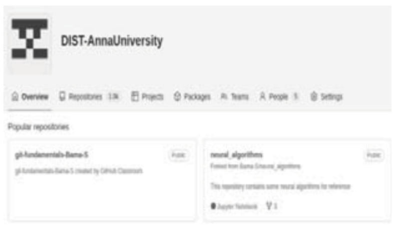
Marks for this submission: 0.00/1.00.

Figure 5 Wrong answer

A. Creation of Classrooms through GHC

To set up a classroom, first we created an organization namely 'DIST-AnnaUniversity' where the repositories of students and teachers could be accommodated, as shown in Figure 6.

,



DIST-AnnaUniversity

Popular repositories

git-fundamentals-Bama-5

Public

git-fundamentals-Bama-5 created by GitHub Classroom

neural_algorithms

Public

This repository contains some neural algorithms for reference

Jupyter Notebook

Figure 6 Creation of Organization in GH

An academic teacher enrolled through GH can create multiple organizations after due verifications from GH. Then we added multiple classrooms, each for a different course linked with the organization that was created earlier. A snapshot of sample classrooms – 'Python Workshop, Machine Learning Lab, Unix Internals and fyp 2022 2023' from our department is shown in
Figure 7. Next, we enrolled students in the specific classroom. GHC provides two methods for

enrolment (i) through LMS such as Moodle (ii) through a manual procedure. Since we had Moodle set up for the course, we connected it with GHC. More than one teacher could be added for a single classroom, which can be done using the “TA and admins” menu.

With this setup in place assignments can be added. We have experimented with two different types of assignment- (i) Programming Courses and laboratories (ii) Project Submissions. These are described in the next two sections.

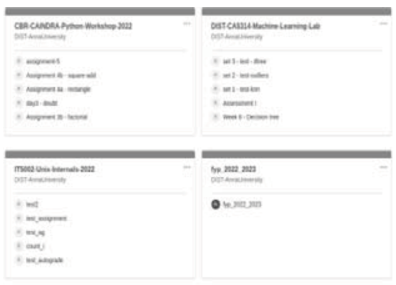


Figure 7 Classrooms created through GHC

B. GHC for Programming Courses and Laboratories

The assignment section of GHC is customized for the programming courses in terms of creation of template repository, assigning each student an individual repository and adding relevant modules and test cases for autograding. These steps are explained below.

1) ***Creation of Template with starter code***: As an initial step, a repository has to be created in an organization as a template. As a running example, here we provide the description of how we created a Python programming exercise with the autograding method for Machine Learning Course. The exercise is to fit a straight line using Linear Regression between the two attributes of the dataset of iris [21]. The actual question that we used is given below.

a) Find the average petal width in the program ‘average.py’. (5 points)

b) Find the range of petal width in the program ‘range.py’. (7 points)

c) In the program ‘regression.py’, fit a straight line between petal length and petal width. Suppose the equation is y = ax+b, find the variables ‘a’ and ‘b’ first. Then for a new petal width value of 5.3, find the predicted value of ‘y’. (13 points)

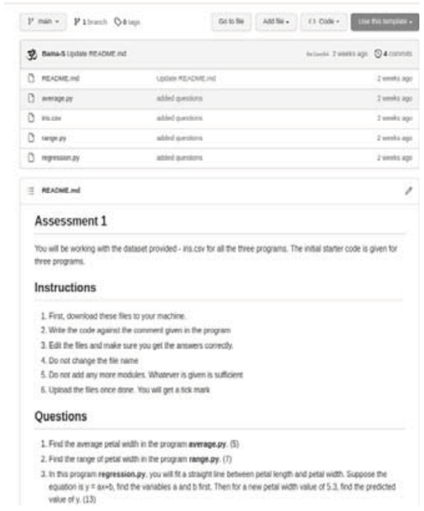
For this specific question, we first created three empty program files namely, ‘average.py’, ‘range.py’ and ‘regression.py’ with the required Python modules. In each of the programs, hints in the form of comments were given, which help the student to add the lines of code at the specific places. For example, the question of regression.py appears as follows. Here, each student can use a different logic to find the values of 'a' and 'b', using which the new predicted value with the variable 'new pl' can be determined.

Question: *#In this program, you will fit a straight line between petal length and petal width. Suppose the equation is y = ax+b, find the variables 'a' and 'b' first. Then for a new petal width value of 5.1, find the predicted value of y.*

```
------------------------------------------------------------
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
#read csv file
#Take X as petal length
#Take y as petal width
#Split train and test with 30% and set random state as 0
#use regression from scikit learn
#find b and round off to 2 decimal places
#find a and round off to 2 decimal places
new_pw = 5.3 #new x value
#calculate y
new_pl = print(round(new_pl,2)) #round off to 2 decimal places
Similar comments were given for program files 'average.py' and 'range.py'.
----------------------------------------------------------------
```

Thus, the question set for this exercise consists of (i) the dataset 'iris.csv' (ii) three python programming files with comments and (iii) readme.md consisting of questions.

In the GHC organization 'DIST-AnnaUniversity' (Figure 6), a repository namely, 'ml-assessment1-2022' for this exercise was created first. The above question set was then pushed/ uploaded and the repository was set as template, as shown in Figure 8. This template was then linked to the assignment section of GHC, which is described next.


Bama-S Update README.md 2 weeks ago 4 commits

README.md	Update README.md	2 weeks ago
average.py	added questions	2 weeks ago
iris.csv	added questions	2 weeks ago
range.py	added questions	2 weeks ago
regression.py	added questions	2 weeks ago

README.md

Assessment 1

You will be working with the dataset provided - iris.csv for all the three programs. The initial starter code is given for three programs.

Instructions

1. First, download these files to your machine.
2. Write the code against the comment given in the program
3. Edit the files and make sure you get the answers correctly.
4. Do not change the file name
5. Do not add any more modules. Whatever is given is sufficient
6. Upload the files once done. You will get a tick mark

Questions

1. Find the average petal width in the program **average.py**. (5)
2. Find the range of petal width in the program **range.py**. (7)
3. In this program **regression.py**, you will fit a straight line between petal length and petal width. Suppose the equation is y = ax+b, find the variables a and b first. Then for a new petal width value of 5.3, find the predicted value of y. (13)

Figure 8 Creation of a template repository

2) ***Adding Assignment in GHC:*** Within each classroom, a new assignment can be created with the options of name, deadline, individual /group assignment, repository visibility with options of private/ public, admin access methods, support of template and editors, addition of test cases and pull request. For the above programming exercise, we have included the following details.

- Name as 'Assessment I"
- Deadline of 90 minutes, which can be altered based on the needs
- Individual assignment, since each student has to work independently
- Private repository
- Provide admin access, so that students have full ownership
- Template link of 'ml-assessment1-2022', which was created as described in Section IV-B1.
- GHC provides codespaces for education purposes, which help students to code within a browser. Hence, we enabled the editor of Codespaces which supports vscode [37].
- Addition of test cases, which is a mandatory step for autograding programming exercises. This is detailed in Section IV-B3
- Pull request, so that teachers can give immediate feedback

Table 1 Test case settings for three python programs

Test Name	Average	Range	Regression
Setup command	sudo -H pip3 install pandas	sudo -H pip3 install pandas	sudo -H pip3 in stall pandas; sudo -H pip3 install scikit-learn; sudo -H pip3 install numpy
Run command	python3 aver age.py	python3 range.py	python3 regression.py
Expected Out put	1.2	2.4	1.85
Comparison	included	included	included
Timeout	10	10	10
Points	5	7	13

3) ***Addition of test cases:*** GHC provides autograding facility through the addition of GH actions. In the assignment options, we can add test case as 'input/output', 'run command' or add specific unit tests for languages python, Java, Node, C and C++, as shown in Figure 9.

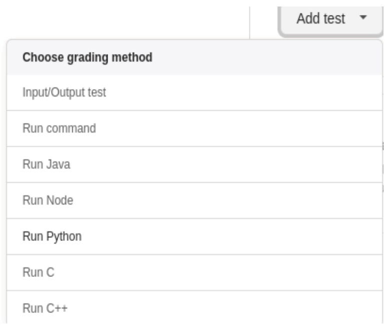


Figure 9 Adding tests for autograding

For the above example, we used 'input/output' option with the settings shown in Table I. The different settings are described below.

- The name of the test, which can be identified distinctly for the three different programs.
- Setup commands are essential, where python libraries are to be included for the execution of programs. The program 'regression.py' (listed above) specifies the libraries of pandas, sklearn and numpy, which needs to be installed before execution of the program. Hence, the setup command should include all the necessary packages required for the program execution.
- Run command is for execution of python programs. Since all the three python programs are of version 3, 'python3 <filename>' is used.

- Expected output of each program is to be given. In the running example, the output values are 1.2, 2.4 and 1.85 for average.py, range.py and regression.py, respectively.
- The comparison has three options (i) included (ii) exact and (iii) regex. Here, we have given 'included' because this particular test case will check for the value which is included in the print statement.
- Default timeout of 10 minutes is given
- Marks (points) for the specific question as 5, 7 and 13 are given for 'average.py', 'range.py' and 'regression.py', respectively.

Alternatively, 'Run Python' option (Figure 9) can also be used, through which 'pytest' is incorporated by GHC. In such cases, the settings and the question type varies. We have also experimented with this type and have found it be useful.

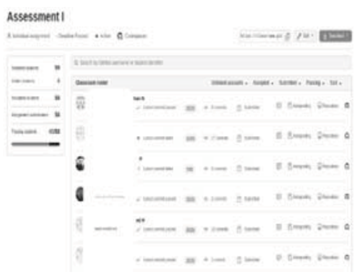
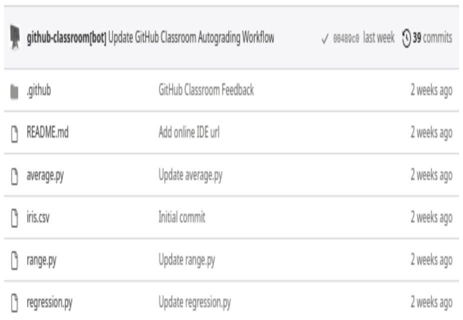

Assessment I

Figure 10 Consolidated student view of assignments

With these settings, the assignment can be created and the link can be distributed to students. Once the student takes up the assignment, the instructor can immediately view the status of students in terms of the number of students who have accepted, submitted and passed with their scores and a link to their repository, as shown in Figure 10. It can be seen that one gets points, only if the test case succeeds in each of the questions. The instructor can view each student's repository to verify the programming structure for the correctness. If the feedback is to be given, it can be done through pull request.

All the above details are described only from the teacher's viewpoint. From the students' perspective, once he/ she accepts the assignment, a unique repository is created by GHC in the same organization with the template where questions are loaded. The student can then download those to their machine, work on it and after ensuring the correctness can upload/ push to their repository. GHC automatically executes the code in the background and checks with the available test cases. This is visible to the student and can be used to debug the code if it fails, as shown in Figure 11. If all the test cases pass, the repository has a green tick mark indicating that the student has successfully completed the exercise, as shown in Figure 12.

Figure 11 Codeexecution at the background



github-classroom[bot] Update GitHub Classroom Autograding Workflow		✓ 08480c0 last week 39 commits
.github	GitHub Classroom Feedback	2 weeks ago
README.md	Add online IDE url	2 weeks ago
average.py	Update average.py	2 weeks ago
iris.csv	Initial commit	2 weeks ago
range.py	Update range.py	2 weeks ago
regression.py	Update regression.py	2 weeks ago

Figure 12 Successful completion

Apart from programming assessments, we also conduct online and hybrid programming workshops through GHC. Recently, we conducted a Python workshop for beginners spanning six continuous days with two hours' online sessions each day for five days and a single day in offline mode. At the beginning of the workshop, participants were neither aware of coding nor comfortable with GH practices, but were able to code and submit programming exercise in GHC within 3 days. Through this method, not only participants get benefited with the coding part, the instructor also can clear the doubts by going through their work and providing the feedback instantly and correcting the codes. The response from participants for this course is given in Figure 13, where 1 means least-effective and 5 means most-effective.

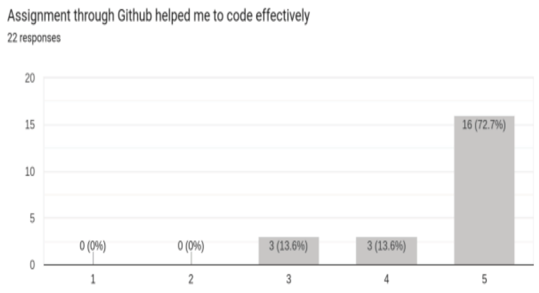
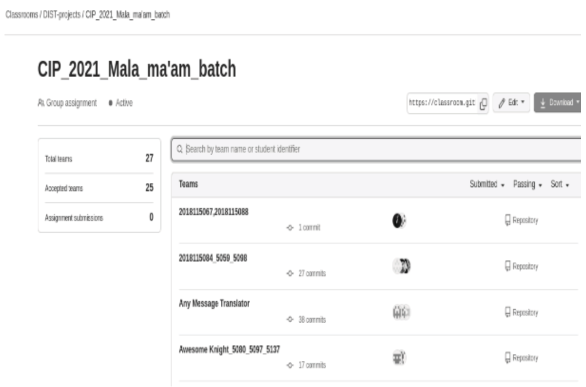


Figure 13 Response of participants from Python Workshop

C. GHC for Project submission

Projects that involve code submissions are encouraged to be submitted through GHC. We have used GHC for project courses such as Final Year Project (FYP), Creative Innovative Project, Socially Relevant Project and Programming with Open Source. For these courses, we use a similar set up of GHC and assignment that was described above, but in a collaborative mode as a group assignment listing the number of members for each team. A sample course with 27 teams is shown in Figure 14. Students, project guide and staff in-charge of project/ course are added as collaborators. While students can work on the code and push these to this repository in a collaborative environment, the staff can verify and provide feedback instantaneously. The project staff in charge can use the number of commits, contributors and frequency of code submission information to evaluate the project. A sample repository of a FYP is shown in Figure 15, indicating the number of commits and the information of codes.


Classrooms / DIST-projects / CIP_2021_Mala_ma'am_batch

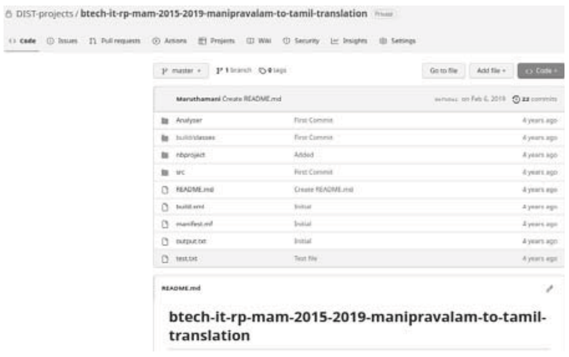
CIP_2021_Mala_ma'am_batch

Group assignment • Active

Total teams	27
Accepted teams	25
Assignment submissions	0

Teams	
2018115067,2018115088	1 commit
2018115084_5059_5098	27 commits
Any Message Translator	38 commits
Awesome Knight_5080_5097_5137	17 commits

Figure 14 Submission of team projects


DIST-projects / btech-it-rp-mam-2015-2019-manipravalam-to-tamil-translation Private

Maruthamani Create README.md on Feb 6, 2019 22 commits

Analyser	First Commit	4 years ago
build/classes	First Commit	4 years ago
nbproject	Added	4 years ago
src	First Commit	4 years ago
README.md	Create README.md	4 years ago
build.xml	Initial	4 years ago
manifest.mf	Initial	4 years ago
output.txt	Initial	4 years ago
test.txt	Test file	4 years ago

README.md

btech-it-rp-mam-2015-2019-manipravalam-to-tamil-translation

Figure 15 Sample FYP team repository in GHC

Thus with GHC, we have automated programming exercises and project submission tasks for the last six years. In the next section, we provide a few insights based on our experiences.

V. Lessons Learnt

Although there are challenges in adapting to the virtual environment, it turns out to be beneficial in the long run. Here, we project some pointers which may help other educational institutions to move towards autograding mechanisms.

- CR is free and open source with zero pricing, which makes it ideal for educational settings.
- GHC is maintained presently by Microsoft and it provides limitless private repositories with other resources for the academic community. These are beneficial because it provides an industry ready environment for practicing computer programming through GH with contribution to open source projects.
- Initially, students have a learning curve to get used to CR and GHC, but after two to three attempts students find it easier and useful.
- For closed book assessment, CR in an intranet setup without internet connection fare better.
- For regular programming exercise and open book assessments, GHC works well owing to the method of question wise test cases, automated grading, and instantaneous feedback from course instructor to students.
- Both CR and GHC supports multiple programming languages and can be used to conduct programming assessment, labs in parallel.
- GHC is suited for both in person labs as well as online sessions.

- Students find this method of programming useful, which enables them to participate in programming contests and industry oriented internship and jobs.
- Training of faculty is essential for both CR and GHC usage in programming courses.
- With GHC, there is always the possibility of copying among students. To eliminate this practice, free plagiarism detection tools such as MOSS [29] can be used. Another approach which we adopted was having random questions distributed to students through Moodle quiz, which significantly reduces the chances of copying.

VI. Conclusion

In this paper, we have described a few automation mechanisms for our programming labs and project submissions. The mechanisms include CR and GHC. We have customized CR based on our course requirements and the major advantage is that a closed book programming assessment could be automated with this setup. We have used GHC for both learning as well as assessments in programming courses through online mode. This automation has multiple advantages from the perspective of students and course instructors. From the student point of view, they can understand the programming constructs better, practice at their own pace and get a real time experience of contribution to the open source community, especially when they use GHC. Instructors save time by eliminating manual evaluation and at the same time help students in learning.

References

1. *Arnow, David, and Oleg Barshay. "WebToTeach: an interactive focused programming exercise system." FIE'99 Frontiers in Education. 29th Annual Frontiers in Education Conference. Designing the Future of Science and Engineering Education. Conference Proceedings (IEEE Cat. No. 99CH37011. Vol. 1. IEEE, 1999.*
2. *B. C. W¨unsche, Z. Chen, L. Shaw, T. Suselo, K.-C. Leung, D. Dimalen, W. van der Mark, A. Luxton-Reilly, and R. Lobb, "Automatic assessment of OpenGL computer graphics assignments," in Proc. 23rd Annu. ACM Conf. Innov. Technol. Comput. Sci. Educ., 2018, pp. 81–86.*
3. *Benford, S. D., et al. 'Ceilidh: A courseware system for the assessment and administration of computer programming courses in higher education." Proceedings of the Interdisciplinary Workshop on Complex Learning in Computer Environments. 1994.*

4. *Choy, Marian, et al. "Experiences in using an automated system for improving students' learning of computer programming." International Conference on Web-Based Learning. Springer, Berlin, Heidelberg, 2005.*
5. *Cipriano, Bruno Pereira, Nuno Fachada, and Pedro Alves. "Drop Project: An automatic assessment tool for programming assignments." SoftwareX 18 (2022): 101079.*
6. *Code Blocks: Open Source, Cross Platform, Free C, C++ and Fortran IDE. Accessed: Jul. 25, 2019. [Online]. Available: http://www. codeblocks. org/*
7. *Coderunner, https://moodle.org/plugins/qtype coderunner*
8. *Code Zinger, https://codezinger.com/pricing/*
9. *D. Thi´ebaut, "Automatic evaluation of computer programs using Moodle's virtual programming lab (VPL) plug-in," J. Comput. Sci. Colleges, vol. 30, no. 6, pp. 145–151, 2015.*
10. *Darwin, Ian F. Checking C Programs with lint. " O'Reilly Media, Inc.", 1988.*
11. *English, John. "Automated assessment of GUI programs using JEWL." ACM SIGCSE Bulletin 36.3 (2004): 137-141.*
12. *Foxley, Eric, et al. "The Ceilidh System An Overview and some experiences of use." Asian Technology Conference in Mathematics, ATCM97, Penang, Malaysia. 1997.*
13. *Foxley, Eric, et al. "The coursemaster automated assessment system-a next generation ceilidh." Computer assisted assessment workshop. 2001.*
14. *G. Singh, S. Srikant, and V. Aggarwal, "Question independent grading using machine learning: The case of computer program grading," in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2016, pp. 263–272.*
15. *Galan, Daniel, et al. "Automated assessment of computer programming practices: The 8-years UNED experience." IEEE Access 7 (2019): 130113-130119.*
16. *Git, https://git-scm.com/*
17. *GitHub Classroom, https://classroom.github.com/*
18. *Github, https://github.com/*
19. *Github, https://github.com/trampgeek/jobe*
20. *H. Kitaya and U. Inoue, "An online automated scoring system for java programming assignments," Int. J. Inf. Educ. Technol., vol. 6, no. 4, p. 275, 2016.*
21. *Iris dataset, https://archive.ics.uci.edu/ml/datasets/iris*

22. *J. Moreno-Le´on, G. Robles, and M. Rom´an-Gonz´alez, "Dr. scratch: Automatic analysis of scratch projects to assess and foster computational thinking," Rev. Educ. Distancia, vol. 15, no. 46, pp. 1 23, 2015.*
23. *Jackson, David, and Michelle Usher. "Grading student programs using ASSYST." Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education. 1997.*
24. *Jupyter autograde, https://pypi.org/project/jupyter-autograde/*
25. *Lobb, R. & Harlow, J. Coderunner: a tool for assessing computer programming skills. Inroads, 7, 2016,47-51.*
26. *Luck, M., and M. Joy. "Computer-based submission and assessment in BOSS." Interactions Online Journal 6 (1999).*
27. *Moodle, https://docs.moodle.org/401/en/Installing Moodle*
28. *Moodle, https://moodle.org/*
29. *MOSS plagiarism checker, https://theory.stanford.edu/~aiken/moss/*
30. *Morris, Derek S. "Automatic grading of student's programming assignments: an interactive process and suite of programs." 33rd Annual Frontiers in Education, 2003. FIE 2003.. Vol. 3. IEEE, 2003. Cheang, Brenda, et al. "On automated grading of programming assignments in an academic institution." Computers & Education 41.2 (2003): 121-131.*
31. *N. A. Rashid, L. W. Lim, O. S. Eng, T. H. Ping, Z. Zainol, and O. Majid, "A framework of an automatic assessment system for learning programming," in Advanced Computer and Communication Engineering Technology. New York, NY, USA: Springer, 2016, pp.967–977.*
32. *NB grader, https://nbgrader.readthedocs.io/en/stable/*
33. *Reek, Kenneth A. "The TRY system-or-how to avoid testing student programs." Proceedings of the twentieth SIGCSE technical symposium on Computer science education. 1989.*
34. *S. Parihar, Z. Dadachanji, P. K. Singh, R. Das, A. Karkare, and A. Bhattacharya, "Automatic grading and feedback using program repair for introductory programming courses," in Proc. ACM Conf. Innov. Technol. Comput. Sci. Educ., 2017, pp. 92–97.*
35. *Saikkonen, Riku, Lauri Malmi, and Ari Korhonen. "Fully automatic assessment of programming exercises." Proceedings of the 6th annual conference on Innovation and technology in computer science education. 2001.*

36. *Truong, Nghi, Peter Bancroft, and Paul Roe. "A web based environment for learning to program." Proceedings of the 26th Australasian computer science conference-Volume 16. 2003.*

37. *Visual Studio Code, https://code.visualstudio.com/*

38. *Von Hagen, William. The definitive guide to GCC. Apress, 2011.*

39. *Y. Wang, H. Li, Y. Feng, Y. Jiang, and Y. Liu, "Assessment of programming language learning based on peer code review model: Implementation and experience report," Comput. Educ., vol. 59, no. 2, pp. 412–422, 2012.*