Dynamic Rate Limiting with TA-Bot in CS1

Jack Forden, Alex Gebhard, and Dennis Brylow
{jack.forden, alexander.gebhard, dennis.brylow}@marquette.edu
Marquette University - Milwaukee, WI

Motivations

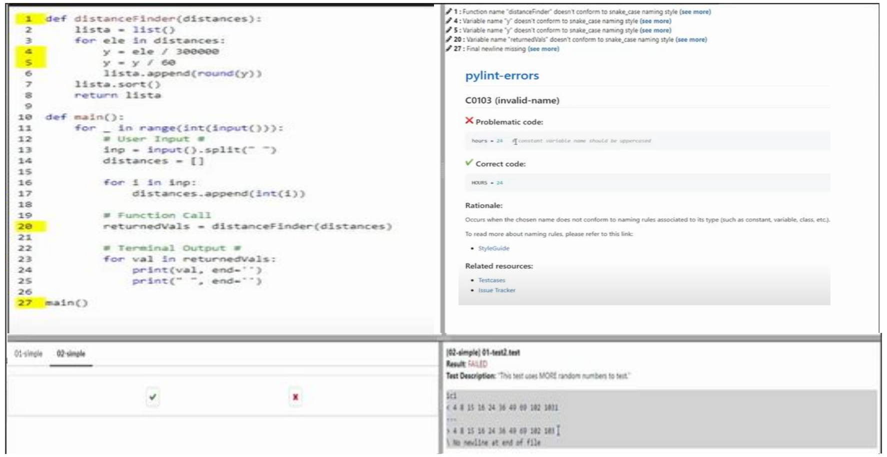
- For instructors:
  - With growing class sizes, instructors have less time to dedicate to individual students.
  - TA-Bot helps instructors track and grade submissions
  - Provides insights into student coding habits, submission rates, and code styles.
- For students:
  - TA-Bot was designed to give automated feedback based on their code
  - Allow students to assess the correctness of their assignment
  - Incentivize students to start assignments earlier

Related Works

- Automated Assessment in Computer Science Education: A State-of-the-Art Review [1]
  - A vast analysis of existing AAT's
- The Marmoset project [2]
  - Used a tokenized rate limiting system
- Can Mobile Gaming Psychology Be Used to Improve Time Management on Programming Assignments? [3]
  - Used an Energy based rate limiting system
  - Authors believed their system was too soft and did not see hypothesized results
- Web-CAT: Automatically Grading Programming Assignments [4]
  - Industry leading Open source AAT
- PyTA: a wrapper for Pylint that provides custom checks for common novice errors as well as improved messages[5]

Creation of Web TA-Bot

- Command line TA-Bot originated from shell scripts
- Web-based TA-Bot uses Flask as it's API framework
- Front end is writing in Typescript with UI components coming from Semantic UI, Javascript, HTML and CSS
- The frontend and backend communicate via a series of JavaScript web requests
- Supports 83 different languages.



```
1   def distanceFinder(distances):
2       lista = list()
3       for ele in distances:
4           y = ele / 300000
5           y = y / 60
6           lista.append(round(y))
7       lista.sort()
8       return lista
9
10  def main():
11      for _ in range(int(input())):
12          # User Input #
13          inp = input().split(" ")
14          distances = []
15
16          for i in inp:
17              distances.append(int(i))
18
19          # Function Call
20          returnedVals = distanceFinder(distances)
21
22          # Terminal Output #
23          for val in returnedVals:
24              print(val, end='')
25              print(" ", end='')
26
27  main()
```

1 : Function name "distanceFinder" doesn't conform to snake_case naming style (see more)
4 : Variable name "y" doesn't conform to snake_case naming style (see more)
5 : Variable name "y" doesn't conform to snake_case naming style (see more)
20 : Variable name "returnedVals" doesn't conform to snake_case naming style (see more)
27 : Final newline missing (see more)

pylint-errors

C0103 (invalid-name)

✗ Problematic code:

```
hours = 24  # constant variable name should be uppercased
```

✓ Correct code:

```
HOURS = 24
```

Rationale:

Occurs when the chosen name does not conform to naming rules associated to its type (such as constant, variable, class, etc.).

To read more about naming rules, please refer to this link:

- StyleGuide

Related resources:

- Testcases
- Issue Tracker

[02-simple] 01-test2.test
Result: FAILED
Test Description: "This test uses MORE random numbers to test."

```
1c1
< 4 8 15 16 24 36 49 69 102 1031
---
> 4 8 15 16 24 36 49 69 102 103
\ No newline at end of file
```

Experiment

TA-Bot implements a novel concept called time between submissions (TBS) to incentivize earlier submissions:

- On the day a project is assigned 5 minute rate limit
- Each day, the rate limit increases, up to two hours on the day the project is due
- Aims to be a more structured rate limit in comparison to Edwards energy system [3]

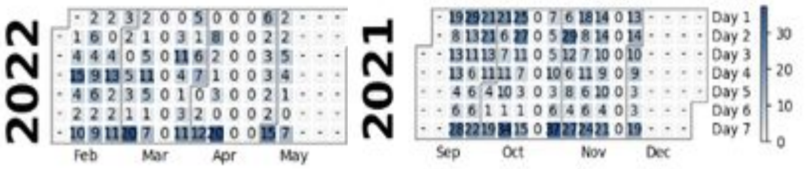
Leveling System and Pylint gamification:

- Test cases for each weekly project separated into different levels
  - Assignments created to better facilitate level system
  - Levels attempt to simulate test driven development (TDD)
    - Base case tests level 1
    - Main functionality tests level 2
    - Edge case tests level 3
- A student could not progress onto the next level, without passing half of current level test cases
- Pylint style suggestions displayed to students
  - Curated Github examples
- A student would be given a "score" on a submission
  - 60 points being allocated to test case results
  - 40 points being allocated to Pylint suggestions
- A student with a score of 70 would be able to unlock a reduced rate limit submission day

Results

Submission heat maps generated for Fall 2021 and Spring 2022 semester.

- Each column represents one week-length assignment. Columns with all zeros represent a week where no homework was assigned.
- **Fall 2021 semester, there is noticeably more students starting the first three days when TBS was enabled**
- The novel concept of TBS did appreciably alter student submission habits in the way it was intended.
- The ability to "unlock" a reduced rate limit day was utilized a handful of times
  - Bad messaging or difficulty using the feature



References

[1]: José Carlos Paiva, José Paulo Leal, and Álvaro Figueira. 2022. Automated Assessment in Computer Science Education: A State-of-the-Art Review.

[2]: Jaime Spacco, Davide Fossati, John Stamper, and Kelly Rivers. 2013. Towards improving programming habits to create better computer science course outcomes.

[3]: Michael S. Irwin and Stephen H. Edwards. 2019. Can Mobile Gaming Psychology Be Used to Improve Time Management on Programming Assignments?

[4]: Stephen H. Edwards and Manuel A. Perez-Quinones. 2008. Web-CAT: automatically grading programming assignments.

[5]: David Liu and Andrew Petersen. 2019. Static Analyses in Python Programming Courses.

Acknowledgements

We would like to thank Tina Boyle Whyte and Justin Wang for allowing us to run this in their CS1 class. We would also like to thank the TAs: John Austin, Nidhi Nellore, and Jiawei Wu. This work is funded in part by NSF grant #CNS-1923597