A Testing Tool for Introductory Programming Courses Thesis B Seminar

Kyu-Sang Kim z5208931

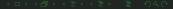
Supervised by Andrew Taylor (UNSW)
Assessed by John Shepherd (UNSW)

Term 2, 2022

yu-Sang Kim Thesis B Seminar Term 2, 2022

Contents

1 Introduction
Thesis Statement
Plan



A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

Develop an extensible and easy to use software package which parses and runs pre-written tests on submitted code

A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

- Develop an extensible and easy to use software package which parses and runs pre-written tests on submitted code
- 2 Implement development procedures that minimise both current and future technical debt

A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

- Develop an extensible and easy to use software package which parses and runs pre-written tests on submitted code
- 2 Implement development procedures that minimise both current and future technical debt
- Remediate known flaws in the existing autotest package

A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

- Develop an extensible and easy to use software package which parses and runs pre-written tests on submitted code
- 2 Implement development procedures that minimise both current and future technical debt
- 3 Remediate known flaws in the existing autotest package
- Maintain backwards compatibility with legacy tests written for the existing autotest package

A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

- Develop an extensible and easy to use software package which parses and runs pre-written tests on submitted code
- 2 Implement development procedures that minimise both current and future technical debt
- 3 Remediate known flaws in the existing autotest package
- Maintain backwards compatibility with legacy tests written for the existing autotest package
- 5 Perform proving and performance tests on the new software package

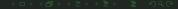
A user-friendly and maintainable general code testing tool is important to streamline the administration of introductory programming courses

We will:

- Develop an extensible and easy to use software package which parses and runs pre-written tests on submitted code
- 2 Implement development procedures that minimise both current and future technical debt
- Remediate known flaws in the existing autotest package
- Maintain backwards compatibility with legacy tests written for the existing autotest package
- 9 Perform proving and performance tests on the new software package
- 6 Deprecate and replace the existing autotest used for introductory programming courses at UNSW CSE

The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module



The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module
 - Implement Core Testcase Parser Module



The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module
 - Implement Core Testcase Parser Module
 - Implement Core Testcase Runner Module

The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module
 - Implement Core Testcase Parser Module
 - Implement Core Testcase Runner Module
 - Run correctness and performance testing on Parser and Runner

The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module
 - Implement Core Testcase Parser Module
 - Implement Core Testcase Runner Module
 - Run correctness and performance testing on Parser and Runner
- 2 Thesis C:
 - Implement Core Testcase Program Correctness Module



The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module
 - Implement Core Testcase Parser Module
 - Implement Core Testcase Runner Module
 - Run correctness and performance testing on Parser and Runner
- 2 Thesis C:
 - Implement Core Testcase Program Correctness Module
 - Implement any extensions that have been deemed necessary by the Design Document

The original plan was set to the following:

- Thesis B:
 - Implement Core Main Module
 - Implement Core Testcase Parser Module
 - Implement Core Testcase Runner Module
 - Run correctness and performance testing on Parser and Runner
- 2 Thesis C:
 - Implement Core Testcase Program Correctness Module
 - Implement any extensions that have been deemed necessary by the Design Document
 - Run correctness and performance testing on complete package and make final adjustments