**Autograder**

**Shane Brandl**

**Bryce Bjorkman**

**David Edwardson**

**Eli Kolb**

**John Snyder**

**Nolan Forehand**

**Tyler Reski**

**Zach Moldenhaur**

**Table of Contents**

[**Installation** 3](#_Toc25591741)

[pyInstall.py 3](#_Toc25591742)

[pyenvsetup.py 3](#_Toc25591743)

[pyreqs.txt 3](#_Toc25591744)

[**File Structure** 4](#_Toc25591745)

[autograder.py 4](#_Toc25591746)

[guiUtils.py 4](#_Toc25591747)

[dynamic\_analysis\_template.py 4](#_Toc25591748)

[tester.py 4](#_Toc25591749)

[commentSummary.py 4](#_Toc25591750)

[**Tutorial** 5](#_Toc25591751)

[**Interacting with Autograder** 6](#_Toc25591752)

[Program Configuration Toggles 6](#_Toc25591753)

[Assignment Key 6](#_Toc25591754)

[File Menu Options 6](#_Toc25591755)

[Import Directory 6](#_Toc25591756)

[Import Zips 6](#_Toc25591757)

[Open Key 6](#_Toc25591758)

[Exit 6](#_Toc25591759)

[**Team Member Contributions** 7](#_Toc25591760)

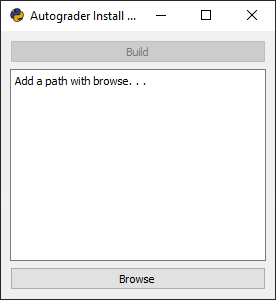
**General**

This project will be developed to assist in the generation of grades for lab assignments in the intro to programming course with Python. This program allows a professor to pass in zip files, directories, and individual files and determines a grade for the submission. This program allows professors to customize grading criteria through a graphical user interface.

# **Installation**

## pyInstall.py

This script will allow for the installation of the Autograder desktop program. Running this script will open a window to select an install directory.



## pyenvsetup.py

This is the environment setup script. It will make sure that the correct version of PyQt5 and other required modules are installed. This script is run by the pyInstall.py script when installing Autograder. If you just wish to run Autograder.py to interact with Autograder, you should run this script before running Autograder.py.

## pyreqs.txt

The following modules are required to run Autograder and are automatically downloaded with the pyenvsetup.py script.

Qt5 version 5.13.1

PyQt5-sip version 4.19.19

python-editor version 1.0.4

pyshortcuts version 1.7

winshell version 0.6

pypiwin32 version 223

pyinstaller version 3.5

# **File Structure**

## autograder.py

Autograder is the GUI that the professor will interact with. It displays options customizing what goes into calculating a student grade. It takes in a professor’s input for script locations and sends each script through dynamic analysis and comment summary.

## guiUtils.py

This script is a utility class that allows for passing information between the interface and the driver.

## dynamic\_analysis\_template.py

Dynamic analysis allows for individual function testing. (TODO)

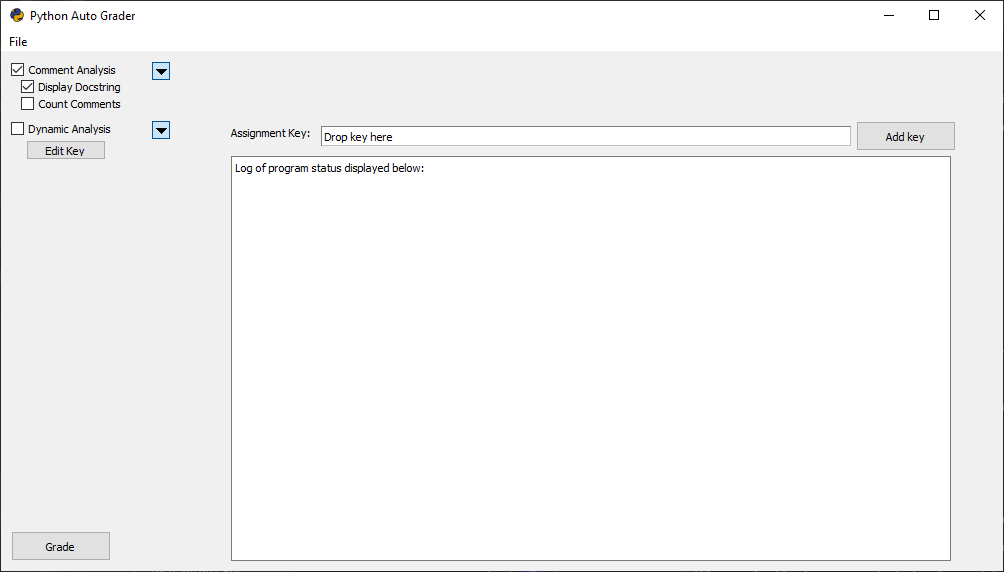
## tester.py

Lorem ipsum. (TODO)

## commentSummary.py

Comment summary finds all docString and inline comments in a script.

# **Tutorial**



Step 1 – Select a Key

Press the “Add key” button and choose the professor’s correct version of an assignment.

Step 2 – Select a Zipped Directory

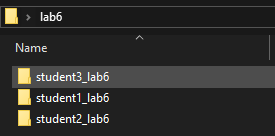
Press the file dropdown and press “Import Zip(s).” Select the desired zipped folders to run against the grading key, and then select the location of where the unzipped version of those folders should be stored. This will **NOT** set the program to grade this newly created folder yet. By default, this directory is named “studentWork.”

Step 3 – Toggle Desired Grading Criteria

Select the boxes for comment analysis and dynamic analysis for the complete Autograder experience.

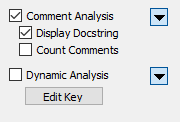
Step 4 – Select Grade

Press the “Grade” button. A new window should pop up that allows you to select a directory. The selected directory should contain subdirectories separated by student. For the following scenario, the “lab6” folder should be selected.



# **Interacting with Autograder**

## Program Configuration Toggles



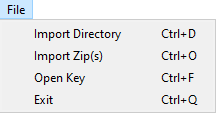
Autograder’s configuration toggles will allow a professor to customize the log output. Checking the box for comment analysis tells Autograder to add comments to the grading criteria.

## Assignment Key



The “Key” is a correct version of a given assignment, likely created by the professor. Keys can be added by either dragging and dropping or selecting the “Add key” button to the right.

## File Menu Options



### Import Directory

Importing a directory will (TODO)

### Import Zips

The import zip(s) selection will first prompt for the location of a single zipped folder or multiple zipped folders. Then it will prompt for the location where the professor would like to store the unzipped folders.

### Open Key

Open key will set the assignment key. The path should show up above the log.

### Exit

Exit will close the program

**Known Bugs**

# **Team Member Contributions**

Shane Brandl (Project Manager)

Arrange Meeting times/Manage Scrum Board

Manage GIT

Front end design

GUI Design and Implementation

Debugging

Bryce Bjorkman

Dynamic Analysis

Py environment setup

Core grading functions

Key editor generator

MOST Back end code design

David Edwardson

SRUM board creation/management

Scope definer

SRS/User Manual

sampleHomework examples

Eli Kolb

Front end GUI design

Pyinstaller executable

Debugging

Key editor link up

Testing

John Snyder

File Menu – Import directories, zips and keys

File Dialogs

Grade Button

GUI Design and Implementation

Nolan Forehand

Comment Summary

User Manual

Code Reviews

Program Icon

Debugging

Tyler Reski

Comment Summary

Separate platform tester (Surface)

Drag & Drop key functionality

Front end GUI design

Zach Moldenhaur

Pyinstaller executable

GUI resizing

GUI Design and Implementation

Debugging