Anna Paulson

May 30, 2020

Foundations of Programming: Python

Assignment 07

<https://github.com/ampaulson/IntroToProg-Python-Mod07>

Error Raising and Pickling: The Bug List

# Introduction

In this paper I will discuss the steps I took to complete Assignment 07 for the class Foundations of Programming: Python. The assignment involves writing a script that demonstrates error handling, or raising, and pickling in python. The script I created is similar to other assignments since it creates and edits a list, but it uses different functions to achieve a similar outcome. Well thought error raising can make or break the program.

# Writing the Script

The script needs to demonstrate two elements: error raising and pickling. I began with pickling. Pickle is a method of opening, reading, writing, and closing external files from a python script by translating file data into binary data in order to reduce the file size. Pickle functions translate the data to binary and then back to English. On the next page, Figure 1 shows the pickle structure used with “bug” variables.

A screenshot of a cell phone

Description automatically generated

***Figure 1. Pickle Functions in Action***

The “Bug List” is opened and edited in the pickle style functions above in figure 1. The .load() function is what translates the binary back to English so that the user can more easily understand the language. I also added a “starter list” with some bugs listed so that there would be more data on the file once the user adds something.

Then I moved on to the error handling portion of the script. I started by adding an error regarding the user’s input. If the user hits enter without typing anything after being prompted, a ValueError will be shown as demonstrated in Figure 2.

A close up of a screen

Description automatically generated

***Figure 2. Value Error***

Figure 2’s ValueError raises to the user that it’s an error to not input any data when prompted.

I continued to add two more errors: the unpickling error and the sys.exit error. The unpickling error comes into play if a user or someone else manually edits the text file that the script reads and writes. The sys.exit error stops execution without printing any traceback and just ends the program. These two errors are shown in the script below in Figure 3.

A close up of a screen

Description automatically generated

***Figure 3. More Errors in the Script***

Any manual edits made to the text file are described to the user in figure 3 as “invalid edits.”

The script was complete and ready to run.

# Running the Script

I began by testing one of the errors by hitting “enter” without inputting any data as shown in Figure 4.

A screenshot of a cell phone

Description automatically generated

***Figure 4. Script Running in PyCharm***

Figure 4 shows the script successfully displayed the error message since no data was entered.

In Figure 5 I demonstrate what happens when a user does input data when prompted.

A picture containing drawing

Description automatically generated

***Figure 5. Script Running without Error in PyCharm***

The script worked properly and allowed pickling and errors to occur. The “Bug List” was created.

Then I checked to see how the script ran in terminal as demonstrated in Figure 6.

A screenshot of a social media post

Description automatically generated

***Figure 6. Script Running in Terminal***

The user’s input in Figure 6 was successfully added to the bug explorer’s list via Terminal.

# Summary

Assignment07 achieved the similar goals as other assignments by creating a list, but it shifted focus to using pickling and errors. The pickling allowed data compression into binary and the error raising allowed for new interaction between the program and the user. Raising an exception can be very beneficial in more complex scripts to keep the program from crashing if there is an issue that needs to be addressed, particularly if effort into the foresight of what might go wrong with the program is completed. I ended up thinking of writing about errors in the script as a form of risk assessment for the program.