Sara O’Neal

May 25, 2021

Foundations of Python Programming (IT FDN 110 A)

Assignment 07

GitHub: <https://github.com/sarao3/IntroToProg-Python-Mod07>

# Pickling and Structured Error Handling in Python

## Introduction

The purpose of this paper is to provide an overview of how to complete the seventh assignment in the Foundations of Python Programming course, which covers how to work with pickling and structured error handling in Python.

## Structured Error Handling in Python

The following steps outline an example of how to use the Try Except block to handle errors within Python.

1. The Try block is the first component, which tests a block of code for errors. In Figure 1, the block of code between lines 49 and 56 will be tested for errors.Graphical user interface, text, application

   Description automatically generated

**Figure 1**: An example of a Try block with code from Module 5.

1. Once the block of code defined with the Try block has been written, a programmer can define within the Except block how to handle and communicate the error. In Figure 2, this can be demonstrated by using a specific built-in Python error called IndexError followed by printing both a custom error message (line 59 in Figure 2) and Python built-in error message (line 61 in Figure 2). Figure 2 also demonstrates that a programmer can include many exception blocks as they want. This is shown by including the non-specific error handling on lines 62 to 65 in Figure 2.

Graphical user interface, text, application

Description automatically generated

**Figure 2:** An example of a Try Except block with both a specific error handling (IndexError) and non-specific error handling (Exception).

## Pickling in Python

The following steps outline an example of how to use the Pickling method within Python. Pickling is a method use in Python to save data in a binary format in order to obscure details of the content and reduce the size of a file. While this method does obscure the contents, it does not encrypt the data.

1. First, import the Pickle module, which will bring in code from another code file.

Graphical user interface, text, application, email

Description automatically generated

**Figure 3:** The pickle module is imported into the current working Python script.

1. In this example, dictionary items are added to a Python object.

Graphical user interface, text, application, email

Description automatically generated

**Figure 4:** The dicObj variable is used to capture the dictionary items within a Python object.

1. The next section of code converts the dicObj Python object into a byte or character stream and saves it to a file called PicklingExample1.dat. The “ab” in the Open function denotes that the data will be appended in a binary format. The pickle.dump() is the function used to serializes the data into the PicklingExample1.dat file.

Graphical user interface, text, application, email

Description automatically generated

**Figure 5:** The object and file are the two parameters passed into the pickle.dump() function.

1. The final section of the illustrates how to reverse what was done in step 3 and read the data back into Python. This is accomplished by using the “rb” when the .dat file is opened and using the pickle.load() function to de-serialize the data back to its original contents.

Graphical user interface, text, application, email

Description automatically generated

**Figure 6:** The pickle.load() function is used to read the data back into Python.

## Summary

This week, we learned how to utilize structured error handling and the pickling method within a Python program. Programmers may find it more beneficial to include custom or specific exception classes within their Python code as a way to identify and resolve errors easier and more efficiently.