**Student Name:** Andy Simpson

**Date:** 3/06/20

**Course:** Foundations Of Programming: Python

**Assignment:** Assignment07

**GitHub Link:** <https://github.com/Asimps2006/IntroToProg-Python-Mod07>

**The “New Student Information Database” Python Script**

**Introduction**

In this module and assignment, I learned about creating Python scripts that use files to store and access information. I also learned how to use “pickle” as a python way to store obscured data in a .dat binary file. Finally, I learned all about how to handle errors that my code might generate.

Specifically, I learned about:

* How to Read data from text files
* How to Write Data from text files
* How to Read/Write more complex data from binary files
* How to intercept and handle errors during a program’s execution

I also learned a bit more about GitHub web pages, how to use create a blog article to demonstrate python programming concepts.

**Working with Text Files:**

I learned on page 190 of Chapter 7, that with Python, it’s easy to read strings from plain text files-files that are made up of only ASCII characters. Text files are a good choice for permanently storing simple information, for a number of reasons. First they are cross-platform. A text file on a windows machine is the same as a text file on a Mac. Second, text files are easy to use. Most operating systems come with basic tools to view and edit them.

**Reading text files:**

Python makes working with files easy, but there may be some confusion because of the different ways you can work with files. For instance, there are several ways to read the data from a file. Here’s a look at some common examples.

The readline() function:

Each time you call the readline() method, gets one line of data and advances to the next line. Advancing one line at a time is commonly referred to in programming as a cursor. Note that since I closed the file, any additional calls to my read\_data() function read the same first row of data!

Using a “while” Loop:

If you want to get data from additional lines you must call the readline() method repeatedly. One way to call the readline() method repeatedly is to use a "while" loop.

The readlines() function:

Python's readlines() function, reads all the lines in a file, and returns a list. The readlines() function is different than the read() function, which reads all the lines in a file and returns a string.

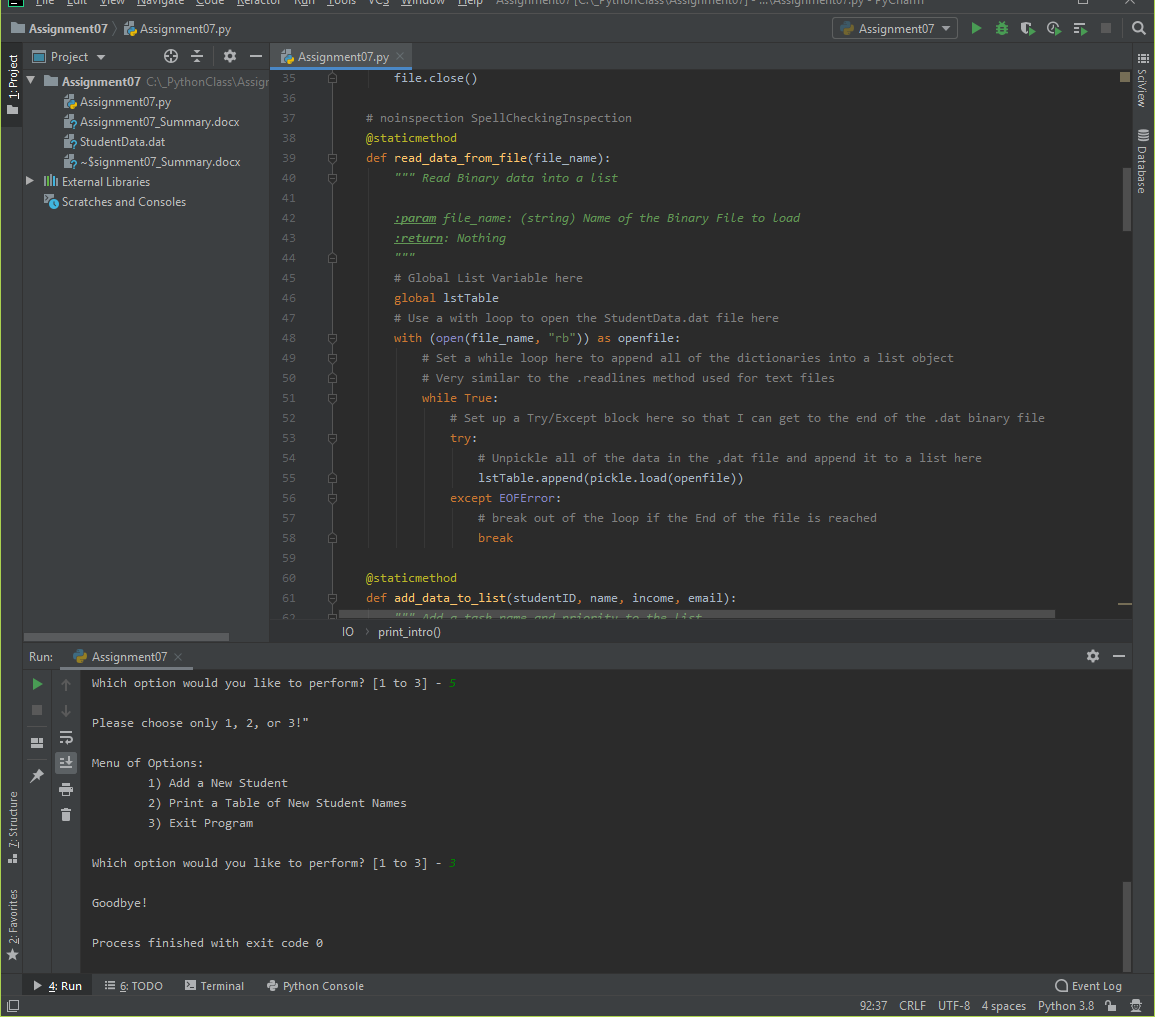
Using a “for” Loop:

Yet another option to read multiple rows of data using a "for" loop. One small advantage of using the “for” loop is that it automatically closes the file when it reaches the end of the file's data.

**Working with Binary Files:**

Data can be saved in binary format instead of just "plain" text. In Python, this technique is called “pickling”. Storing data in a binary format can obscure the file's content and may reduce the file's size.

Important: While the file's content may be more difficult for humans to read, it is not encrypted. So, do not save sensitive data in a binary file and think it is secure!



**Figure 1** *– Example of Reading “Binary” data using python pickle (from Assignment07)*

**Structured Error Handling (Try-Except):**

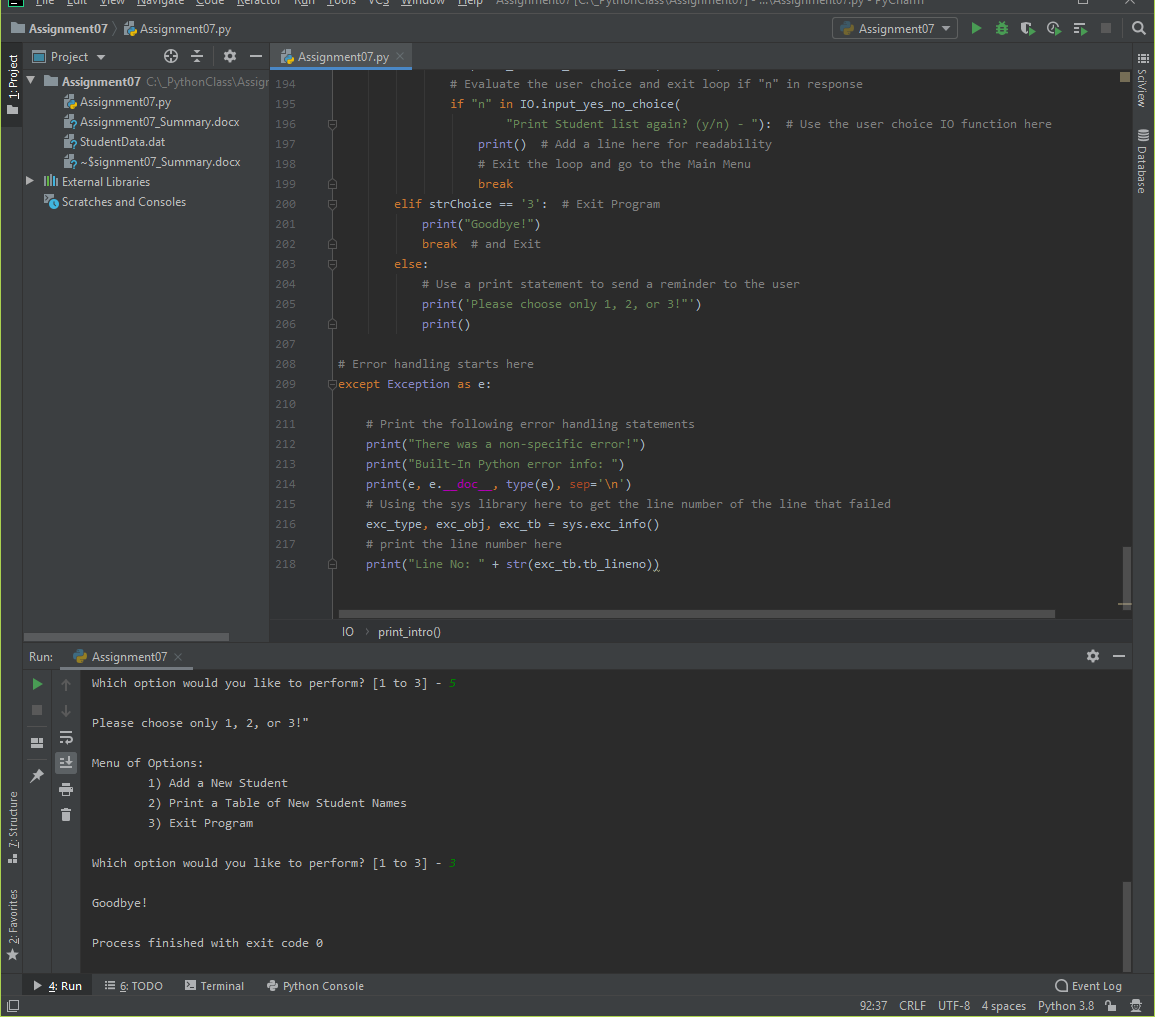
When you are programming, you fix your bugs immediately and make sure the code runs smoothly. However, it often happens that other people introduce new bugs when they use your program. For example, they may change the name of a data file, causing the file not to be found, or input data that does not fit well with your program's design.

You can trap these errors in your programs using a try-except block of code. Doing so allows you to customize how your program handles errors instead of just letting Python do that for you. It is a good idea to add a try-except block to your programs whenever you think human interaction might cause a problem (figure 1 above).

**Using the Exception Class:**

"Exception" is a built-in python class used to hold information about an error. Python automatically creates an Exception object when an error occurs. The Exception object automatically fills with information about the error that caused the exception.

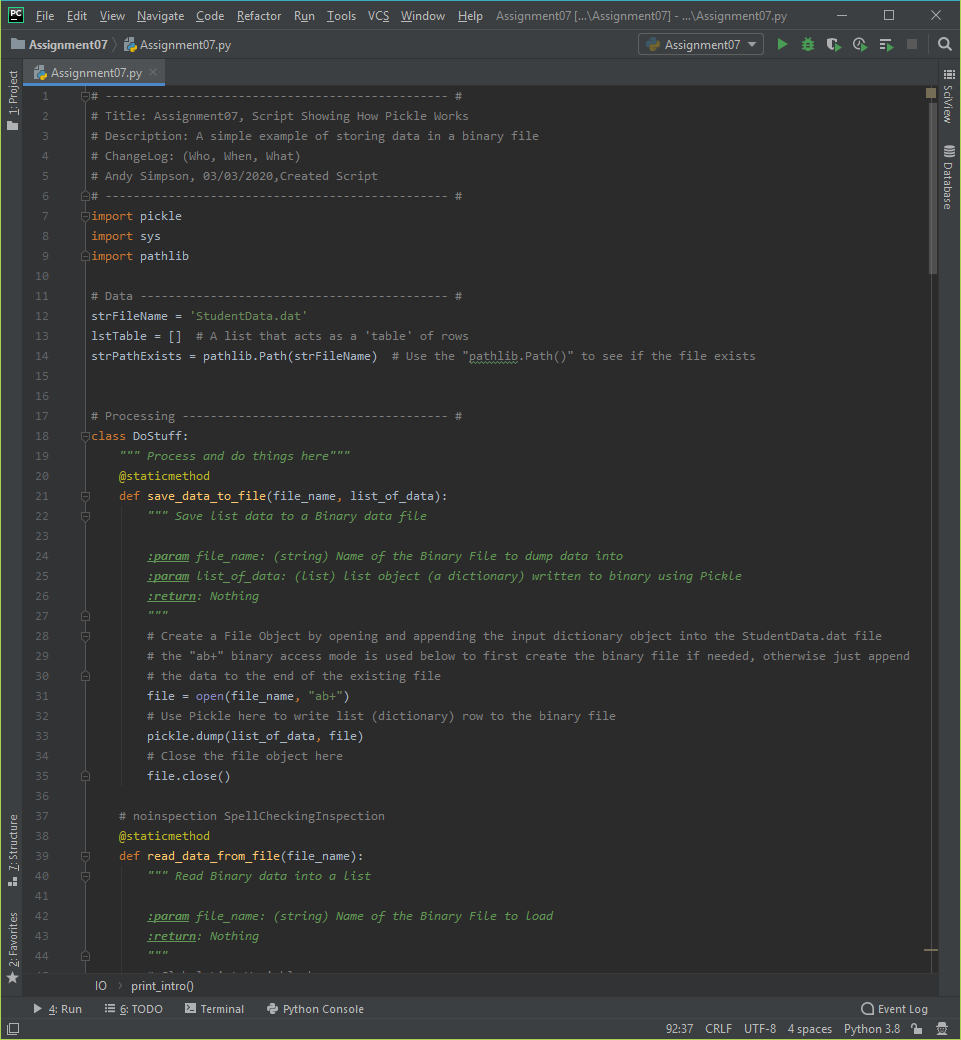
You can capture the Exception object in the except section of a try-except block and extract the error messages as shown below in Figure 2.



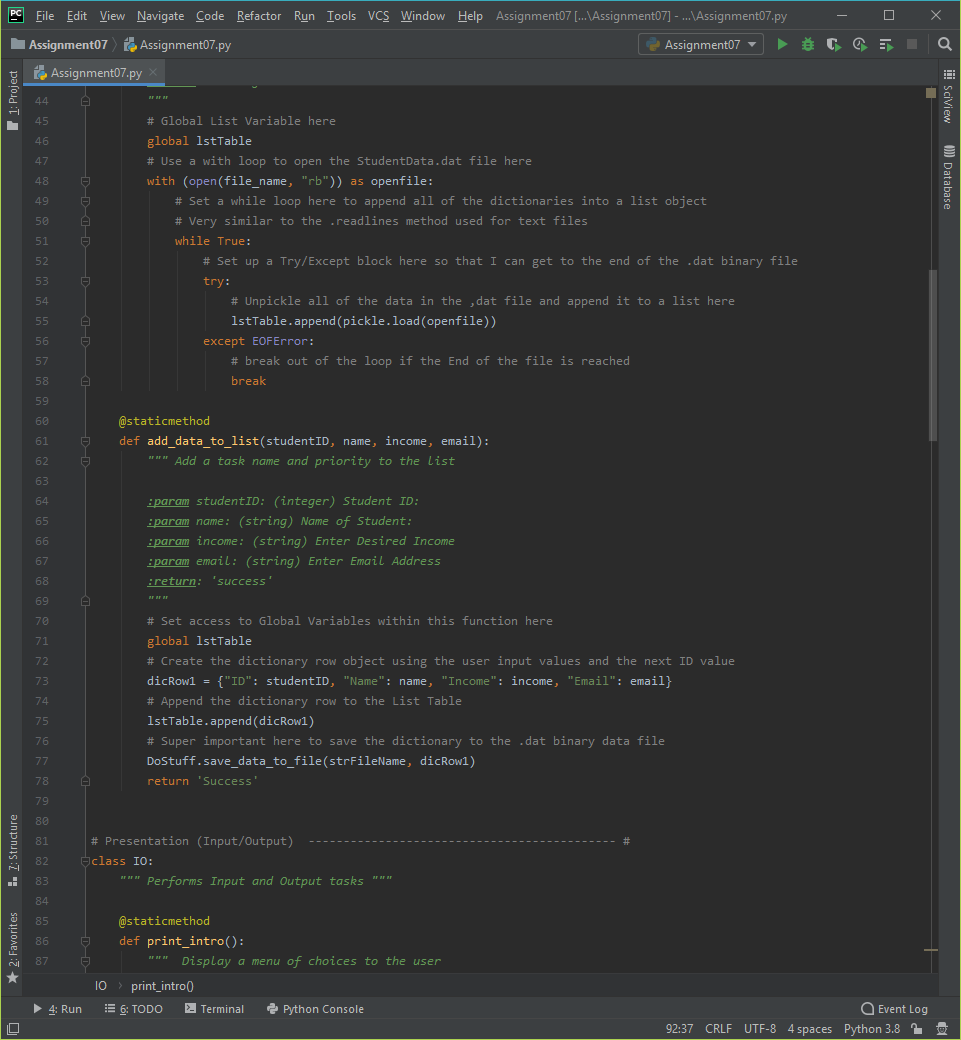
**Figure 2** *– Example of using the Python Exception Class (from Assignment07)*

**Assignment07 – The “New Student Information Database” Script:**

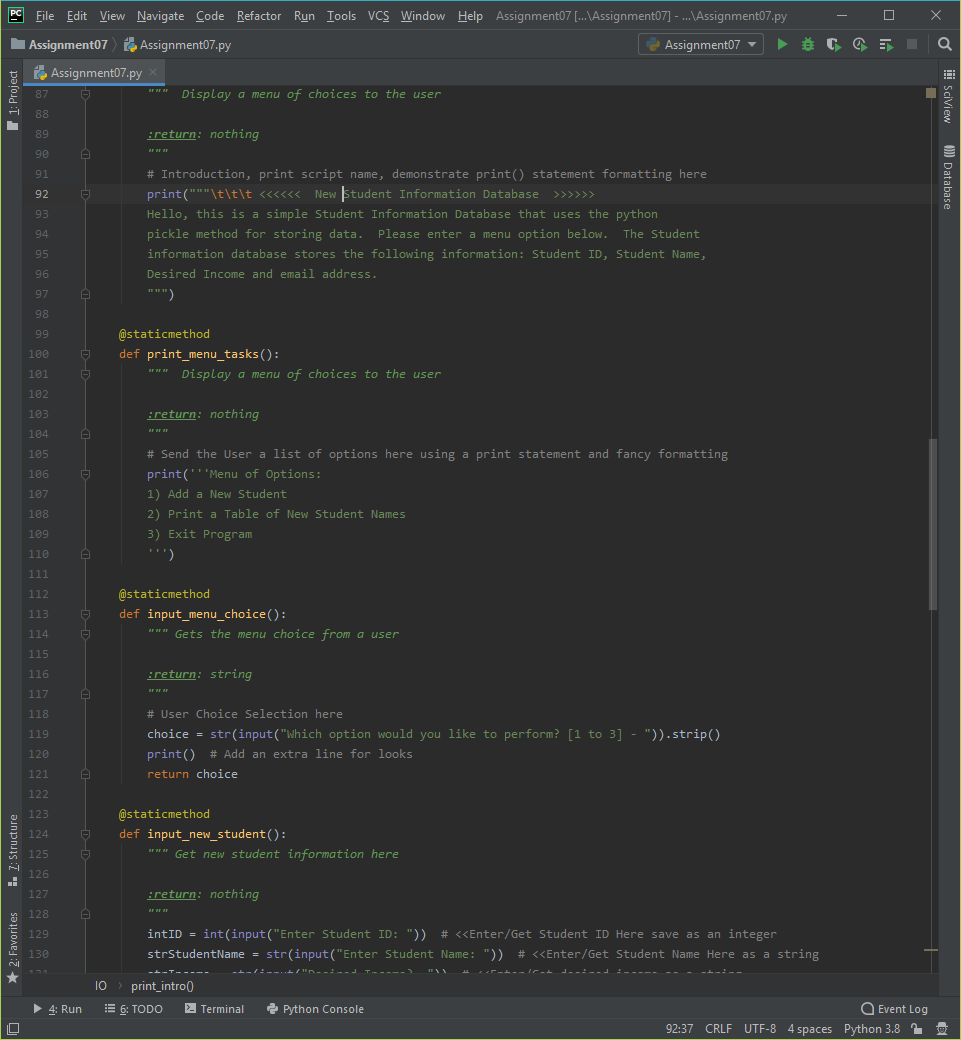
Figures 3, 4, 5, 6 and 7 show the code from the “New Student Information Database” Python Script. Figures 8 and 9 show this script running from within the PyCharm IDE. And then Figures 10 & 11 show the script running from the Command Console.



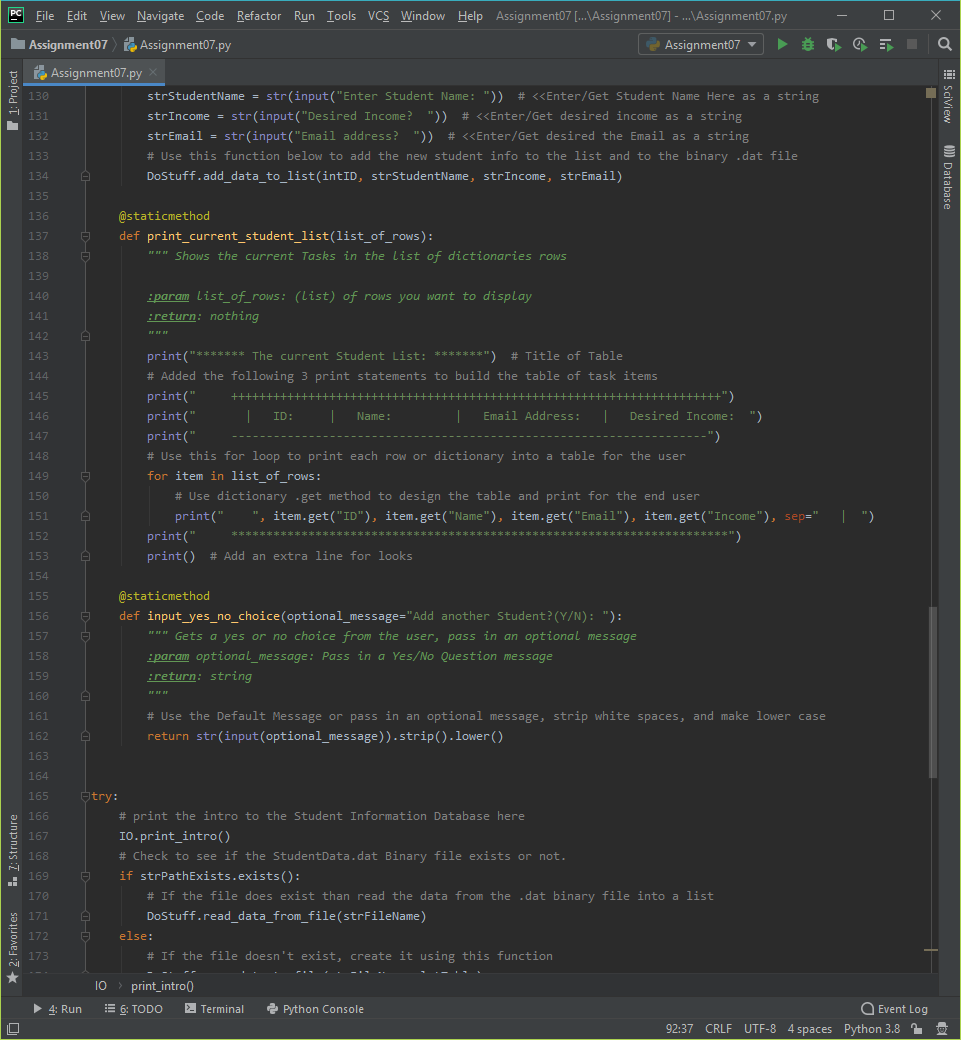
***Figure******3*** *– “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*



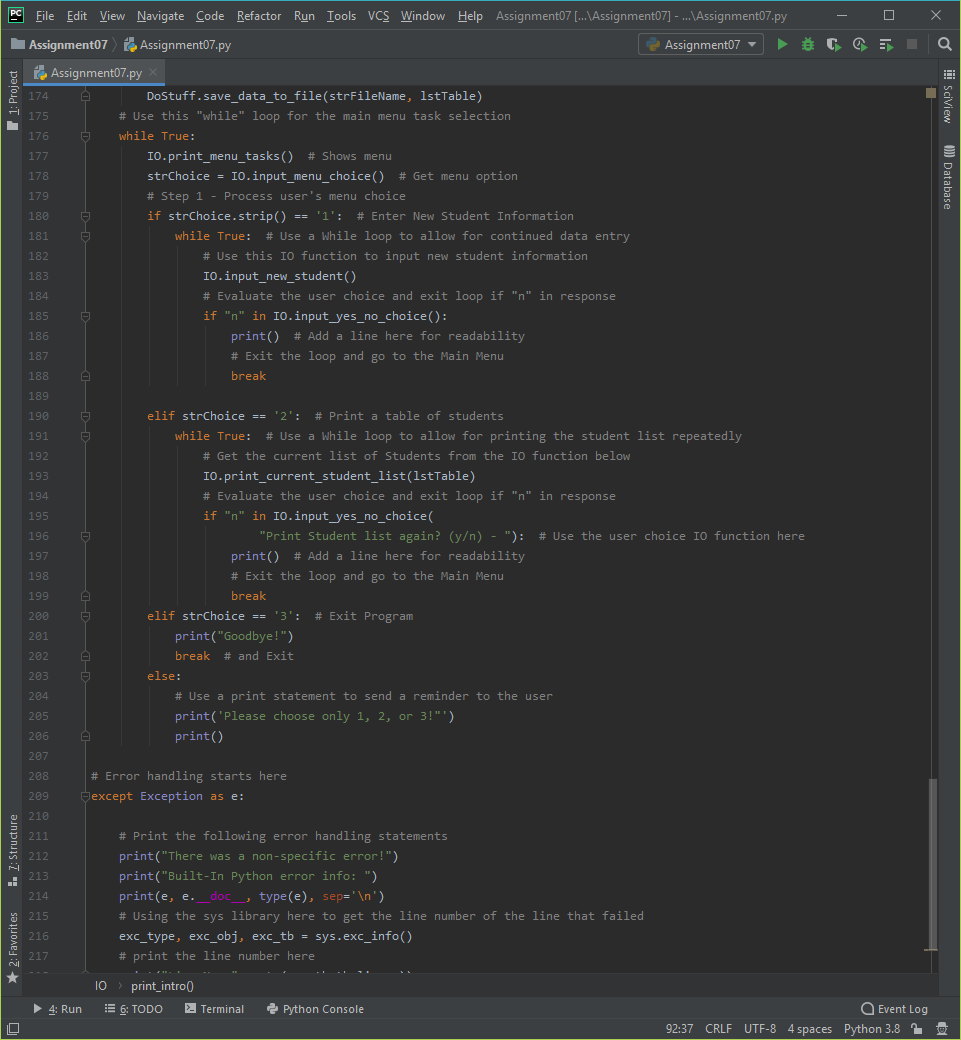
***Figure******4*** *– “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*



***Figure******5*** *– “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*

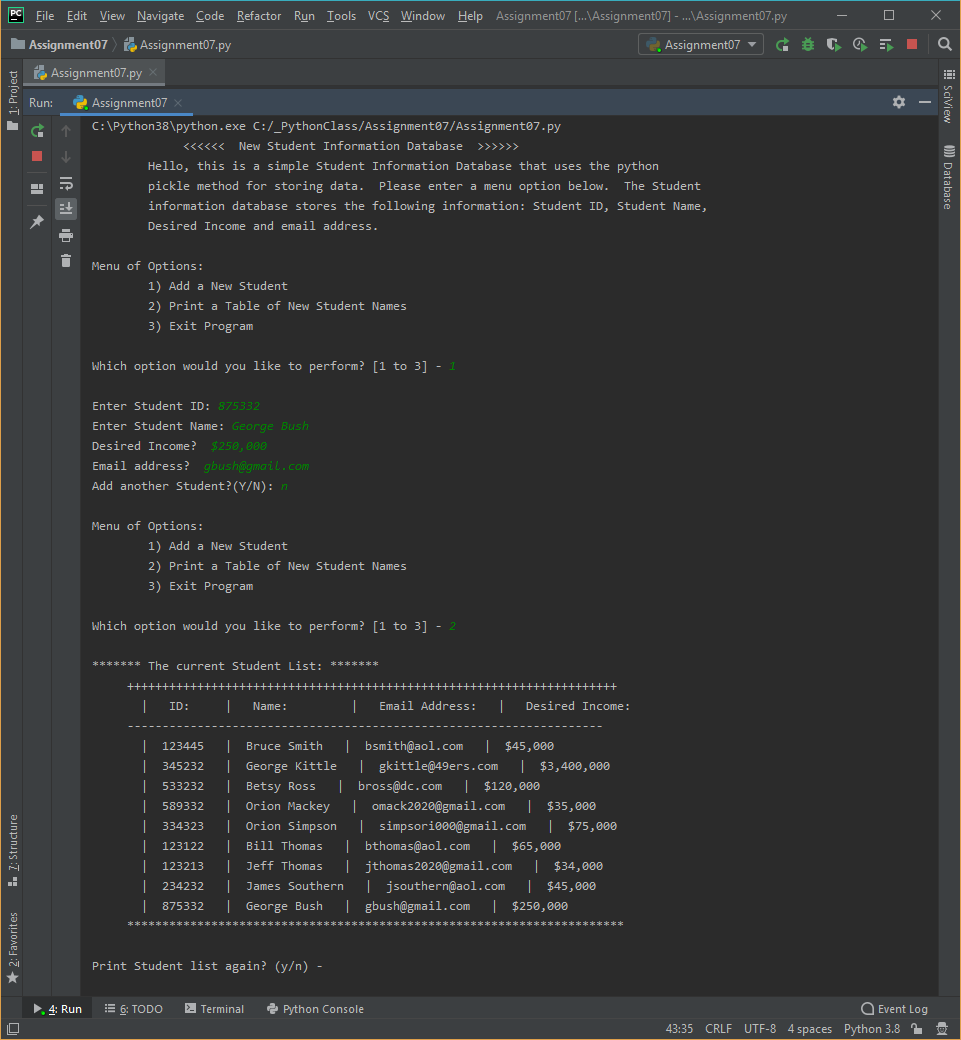


***Figure******6*** *– “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*

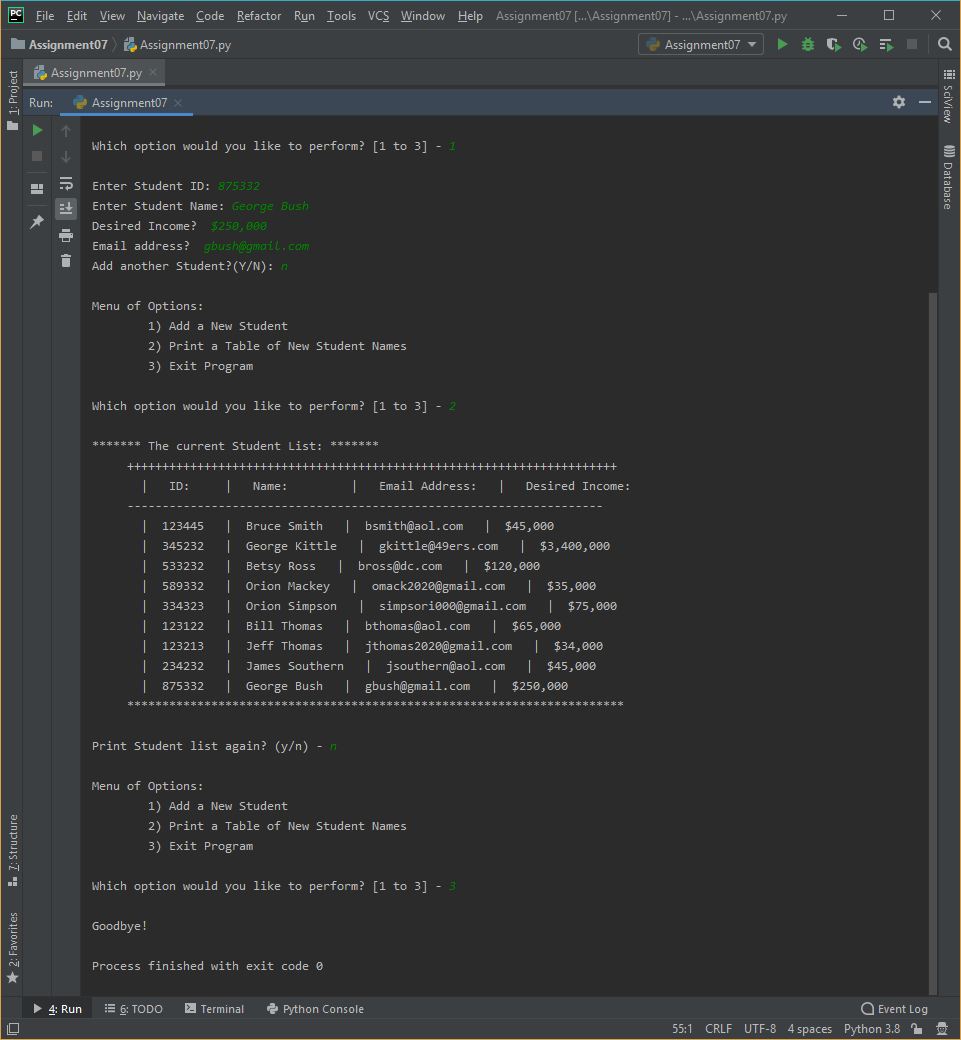


***Figure******7*** *– “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*

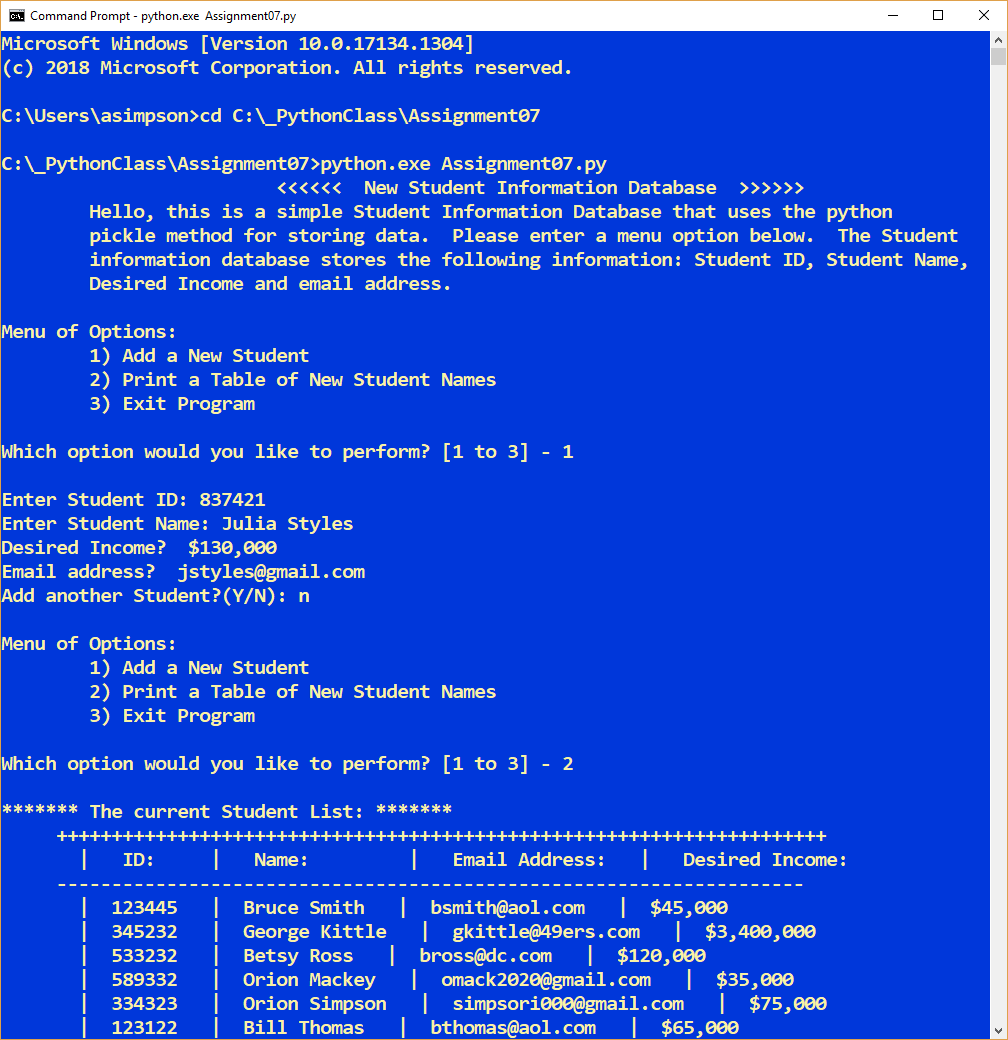
The following screenshots, Figures 14, 15 and 16 shows the Assignment 4 “Home Inventory” script running from within the PyCharm IDE. Please note that I added an introduction to the script as well as a message informing the user that the script has closed.



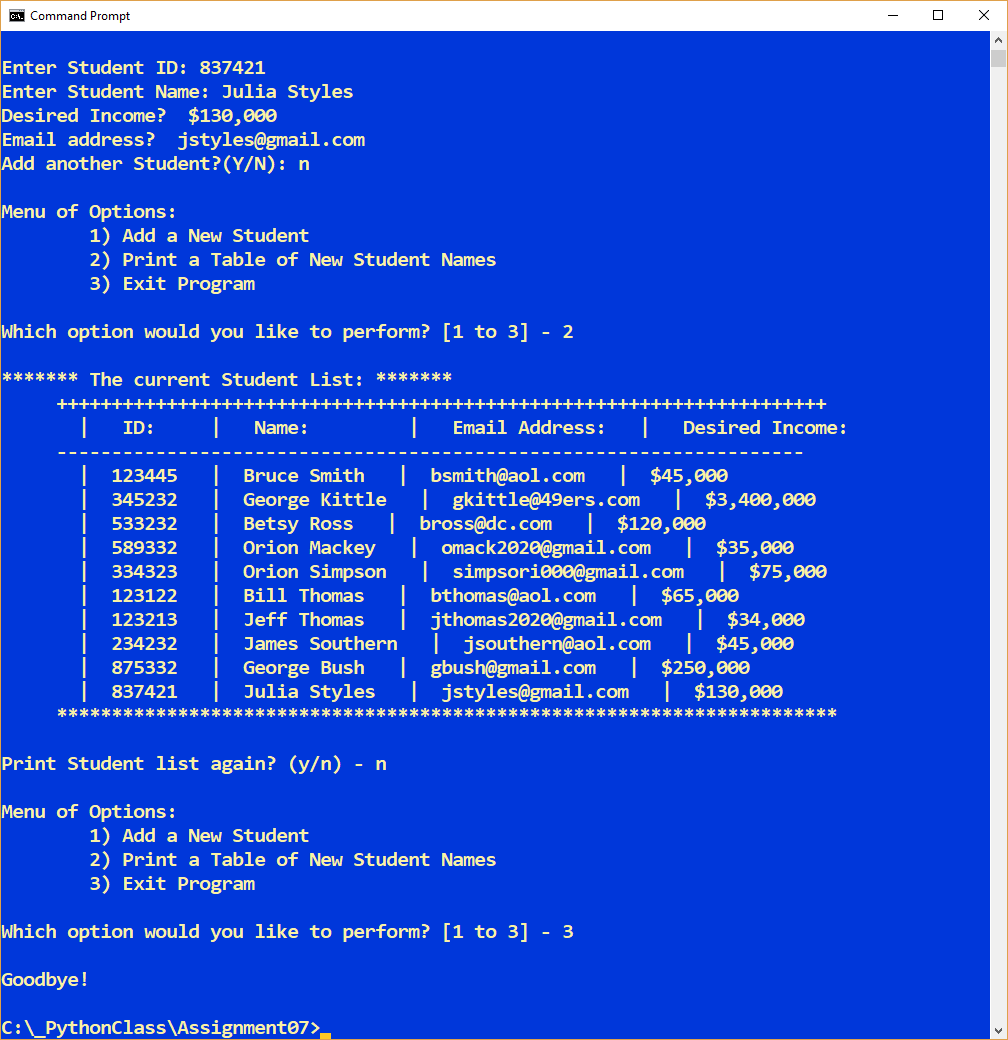
***Figure******8*** *– Running Option #1 & 2 from the “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*



***Figure******9****– Running Option #1, #2 and #3 from the “New Student Information Database” Python Script (Assignment 7) using PyCharm IDE*



***Figure******10*** *– Running Option #1 & 2 from the “New Student Information Database” Python Script (Assignment 7) using the Command Console*



***Figure******11****– Running Option #2 and #3 from the “New Student Information Database” Python Script (Assignment 7) using the Command Console*

**Summary**

In summary, as part of module 7 and assignment 7, I learned all about files and exceptions.

I learned how to read from text files, how to read a single character or an entire file all at once. I learned several different ways to read one full line at a time, which is probably the most common method to read a text file. I also learned how to write to text files, everything from a single character to a list of strings.

I also learned about how to save more complex data to files through pickling and how to manage a group of pickled objects. Finally, I learned about error handling and how to deal with exceptions raised during the execution of a program.

In conclusion, I put many of the concepts mentioned above into the “New Student Information Database” python script. I’m ready to take on chapter 8 where we’ll learn about Object-Oriented programming (OOP).