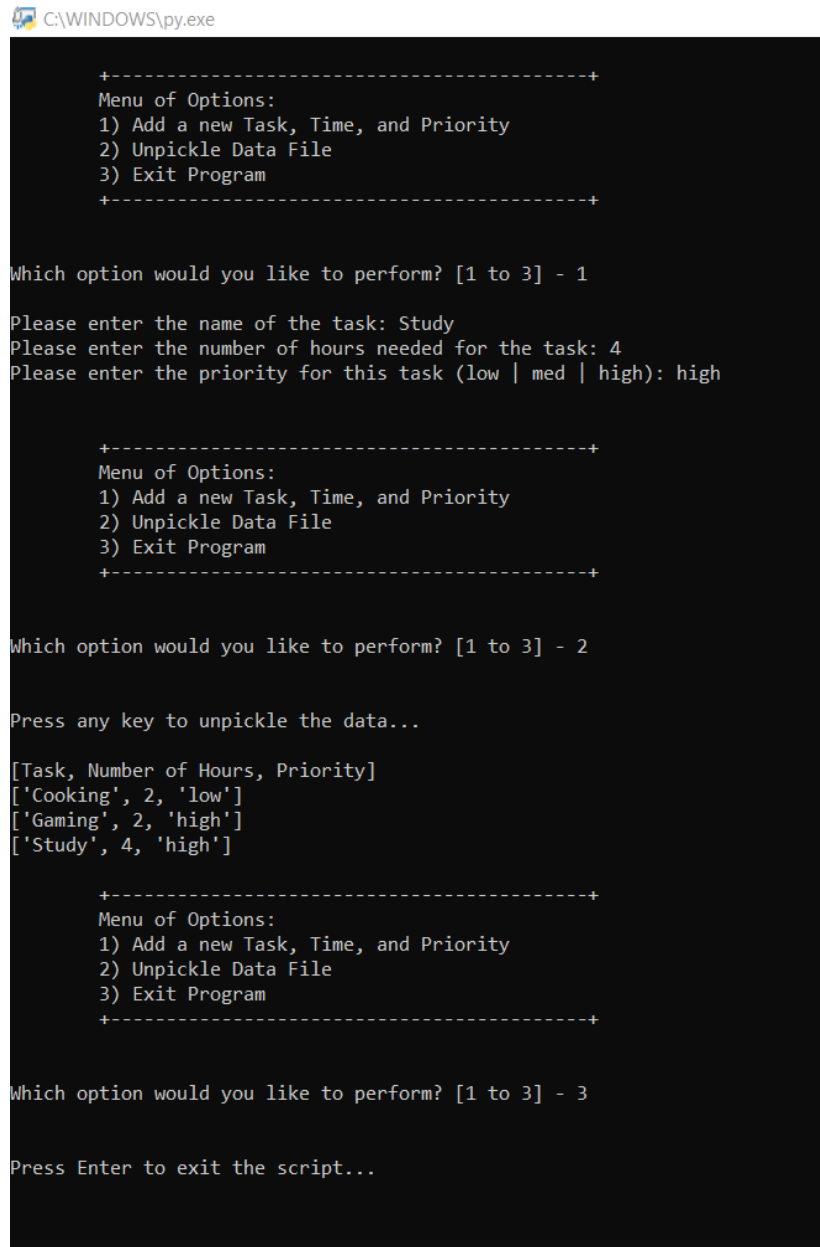


## “ToDoFile” Python Script



```
C:\WINDOWS\py.exe

+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 1

Please enter the name of the task: Study
Please enter the number of hours needed for the task: 4
Please enter the priority for this task (low | med | high): high

+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 2

Press any key to unpickle the data...

[Task, Number of Hours, Priority]
['Cooking', 2, 'low']
['Gaming', 2, 'high']
['Study', 4, 'high']

+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 3

Press Enter to exit the script...
```

**Figure 1.1: Seventh Assignment: Working with Exception Handling & Pickle/Unpickle.**

---

## Introduction

The objective of this assignment is to incorporate **exception handling** and a new method of saving data by using **pickle/unpickle** into what we have been learning in previous weeks. Quite different from the previous weeks, we now have to do our own research from online sources and apply what we learn to a new script.

## Learning Exception Handlings

Usually, when we run into syntax errors in the codes, the program will stop its execution and terminate the script itself. By introducing Exception Handling into our script, the codes will change the normal flow of our program instead of shutting itself down. In order to use this method, we will be using the following keywords: try, except, else, and finally. The keywords function as below:

### Syntax:

```
try:
    # Some Code....

except:
    # optional block
    # Handling of exception (if required)

else:
    # execute if no exception

finally:
    # Some code .....(always executed)
```

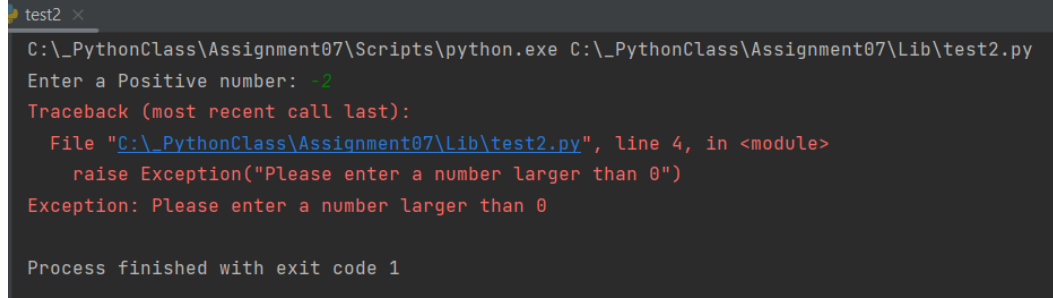
**Figure 2.1: Keywords that can be used in Exception Handling.**

---

There is also a way for us to force a specific exception to occur by using the **raise** keyword. When using raise, the programmer can define what kind of error the program would raise for a particular exception. It is usually accompanied by printed texts to signify errors.

```
x = int(input("Enter a Positive number: "))

if x < 0:
    raise Exception("Please enter a number larger than 0")
else:
    input("Press Enter to exit the program.")
```



```
test2 x
C:\_PythonClass\Assignment07\Scripts\python.exe C:\_PythonClass\Assignment07\Lib\test2.py
Enter a Positive number: -2
Traceback (most recent call last):
  File "C:\_PythonClass\Assignment07\Lib\test2.py", line 4, in <module>
    raise Exception("Please enter a number larger than 0")
Exception: Please enter a number larger than 0

Process finished with exit code 1
```

**Figure 2.2: A simple example of using the “raise” keyword.**

Learning Materials Source:

(geeksforgeeks, <https://www.geeksforgeeks.org/python-exception-handling/> , 2022)

## Learning Pickle/Unpickle:

The pickle module allows programmers to save data to a binary .dmp file and unpickle would reverse the process and display the data so that the user can see it. It is somewhat similar to read and write text files we have been learning but this new module apparently saved the file into a .dmp file for transporting data purposes. There are 2 new functions introduced, which are dumps() and loads(). These will be implemented in the actual assignment so more to that later.

Learning Materials Source:

(geeksforgeeks, <https://www.geeksforgeeks.org/python-exception-handling/> , 2022)

---

## Creating the Script with PyCharm

As always, a simple Script Header is used to provide a few descriptive details about the program as well as declared extra variables that would be used for the script. Since we will be using the pickle module for the first time, we will import the whole one, which seems to already exist within PyCharm.

```
# ----- #
# Title: Assignment 07
# Description: Working with Exception Handling and Pickle/Unpickle data.
# ChangeLog (Who,When,What):
# Vu Thai,11/29/2022,Created and Modified code to complete assignment 07
# ----- #

import pickle #imports the pickle module

lstPickle = []
strFile = "ToDoList.dmp"
```

**Figure 3.1: Script Header and importing pickle module.**

As I always want to incorporate what I have learned from the previous week into the new project, I went ahead and declared two functions for Menu and Choice Selection.

```
class IO:

    @staticmethod
    def output_menu_tasks():
        print('''
+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+
''')
        print() # Add an extra line for looks

    def input_menu_choice():
        choice = str(input("Which option would you like to perform? [1 to 3] - ")).strip()
        print() # Add an extra line for looks
        return choice
```

**Figure 3.2: Two functions were created for the menu and selection.**

Next, I went ahead and sent the previous two methods into a **while loop** so the user can interact with the script. This method was introduced weeks ago but I found it very useful and fun to use!

```
while (True):
    # Step 3 Show current data
    IO.output_menu_tasks() # Shows menu
    choice_str = IO.input_menu_choice() # Get menu option

True) > if choice_str.strip() == '1' > while (True) > try
test x
C:\_PythonClass\Assignment07\Scripts\python.exe C:\_PythonClass\Assignment07\Lib\test.py

+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+
```

**Figure 3.3: Using while loop to display the menu.**

As for option 1, the user will be asked to enter a Task, Time, and Priority similar to the previous assignment. To add a little bit of a new twist to the assignment, there is a new input that will ask for the approximate time needed to complete the task. This will create a good opportunity for me to use exception handling to force the user to enter an actual number only!

```
while (True):
    try:
        intTime = int(input("Please enter the number of hours needed for the task: "))
    except ValueError:
        print("Please use whole number integer only!") # If anything other than an integer is returned print this message
        continue
    else:
        break

0) > if choice_str.strip() == '1' > while (True) > except ValueError
test x
+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 1

Please enter the name of the task: task
Please enter the number of hours needed for the task: 1
Please use whole number integer only!
Please enter the number of hours needed for the task:
```

**Figure 3.4: Using exception handle to force the user to input number only.**

---

Next, I tried the raise method to create another exception handler to make the user input low, med, or high only for priority but it didn't seem to work out. I attended the TA session and after a while, we decided it is best to use a list and if-else statement for this purpose.

```
while (True):
    lstCheck = ["low", "med", "high"]
    strPriority = input("Please enter the priority for this task (low | med | high): ").strip().lower() # Use
    if strPriority.lower() in lstCheck:
        print()
    else:
        print("Please enter 'low', 'med', or 'high' only")
        continue
    break
lstToDo = [strTask, intTime, strPriority] # Task, time, and priority are placed into a list for pickling
continue # to show the menu

> if choice_str.strip() == '1' > while (True) > except ValueError
test x
1) Add a new task, time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 1

Please enter the name of the task: Sleep
Please enter the number of hours needed for the task: 0
Please use whole number integer only!
Please enter the number of hours needed for the task: 4
Please enter the priority for this task (low | med | high): undefined
Please enter 'low', 'med', or 'high' only
Please enter the priority for this task (low | med | high):
```

**Figure 3.5: Combining list and if-else to force the user to input the correct option.**

Next, for option 2, here is where the script will pickle and unpickle the input data into a dmp file and read it upon selecting this option. Since most of the actual processes are being handled invisibly, the only display the user can see is the data inside the dmp file after it is done unpickling. Similar to how we created a new text file, this pickle module creates a dmp file instead.

Source:

(geeksforgeeks, <https://www.geeksforgeeks.org/pickle-python-object-serialization/> , 2022)

```

elif choice_str == '2': # Pickle and Unpickle data
    # Store inputs into a pickle dump file
    objFile = open(strFile, 'ab') # Create file and opens it in append mode
    pickle.dump(lstToDo, objFile) # Dump the list data to the file
    objFile.close() # Close the file when done writing
    strUnPickle = input("\nPress any key to unpickle the data...\n")
    objFile = open(strFile, 'rb') # Read the contents of the dump file
    while True:
        try:
            pkldata = pickle.load(objFile)
            lstPickle.append(pkldata) # Append it to list
        except EOFError:
            break

    # Close the file
    objFile.close()
    print("[Task, Number of Hours, Priority]")

    # Traverse the data list, printing display
    for pkldata in lstPickle:
        print(pkldata)

    continue # to show the menu

```

**Figure 3.6: Saving and Displaying data with pickle and unpickle.**

```

+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 2

Press any key to unpickle the data...

[Task, Number of Hours, Priority]
['Cooking', 2, 'low']
['Gaming', 2, 'high']
['Study', 4, 'high']
['Sleep', 4, 'high']

```

**Figure 3.7: Unpickling being done on PyCharm.**

---

And again, option 3 would be used to exit the program when selected.

```
elif choice_str == '3': # Save Data to File
    strExit = input("\nPress Enter to exit the script...\n")
    break # to show the menu
```

```
+-----+
Menu of Options:
1) Add a new Task, Time, and Priority
2) Unpickle Data File
3) Exit Program
+-----+

Which option would you like to perform? [1 to 3] - 3

Press Enter to exit the script...

Process finished with exit code 0
```

**Figure 3.8: Option 4 to exit the program.**



---

## Conclusion

Overall, this assignment was a bit different compared to other assignments. I enjoyed my spent time researching and learning about Exceptional Handling and Pickle/Unpickle. However, Exceptional Handlings seems to be only useful when dealing with read/write data into files like what we have been doing. Through this assignment, I was able to see only three different kinds of errors, which are `KeyError`, `ValueError`, and `ZeroDivisionError`. And as for Pickle/Unpickle, I really don't see the importance of using this method but it was interesting to learn that there are various different techniques to manipulate and save data.