# ------------------------------------------------------------------------ #

# Title: Assignment 07

# Description: Research Exception Handling & Pickling in Python

# ChangeLog (Who,When,What):

# Kstevens,11-20-19,Modified code to complete assignment 7

# ------------------------------------------------------------------------ #

# Research source: Python Pickle Module for saving objects (serialization), #https://pythonprogramming.net/python-pickle-module-save-objects-serialization/

# Research source (other/related to above): 11.1. pickle — Python object serialization, #https://docs.python.org/2/library/pickle.html

# Research topic: Pickle Module

# Code Version 7.9

# Description: This tutorial is going to cover the pickle module, which is a part of your standard library with your installation of PythonExample:

Example:

# ( Code implementation can be followed along in the YouTube tutorial, [Python Pickle Module for saving objects (serialization)](https://www.youtube.com/watch?v=2Tw39kZIbhs), (external link).)

import pickle

example\_dict = {1:"6",2:"2",3:"f"}

pickle\_out = open("dict.pickle","wb")

pickle.dump(example\_dict, pickle\_out)

pickle\_out.close()

First, import pickle to use it, then we define an example dictionary, which is a Python object.  
Next, we open a file (note that we open to write bytes in Python 3+), then we use pickle.dump() to put the dict into opened file, then close.

The above code will save the pickle file for us, now we need to cover how to access the pickled file:

pickle\_in = open("dict.pickle","rb")

example\_dict = pickle.load(pickle\_in)

Open the pickle file  
Use pickle.load() to load it to a var.

That's all there is to it, now you can do things like:

print(example\_dict)

print(example\_dict[3])

This shows that we've retained the dict data-type.

Through saving the serialized object, it's nature is included, so we don't have to worry about loading things "as" strings, dictionaries, lists, etc.