# Exam1-Spring2021-Deploy

August 17, 2021

## 0.1 # ENGR 1330 Exam 1 Sec D01/D04 Spring 2021

Instructions: - Work directly in this notebook, upon completion render the notebook as a PDF file. - Upload your solution PDF **AND** the .csv and .ipynb (3 files) to the Blackboard repository for grading! - Multiple attempts are allowed in the 24-hour time window - Internet resources are allowed, remember to cite what you copy from elsewhere. - Consulting with each other is not permitted. - Sharing work with each other is not permitted. - Working together is not permitted.

---

## 0.2 Full name

## 0.3 R#:

## 0.4 HEX:

## 0.5 ENGR 1330 Exam 1 Sec 003/004

## 0.6 Date:

## 0.7 Question 1 (1 pts):

Run the cell below, and leave the results in your notebook.
If you get an ERROR message, leave it and continue.

[1]:

```python
#### RUN! this Cell ####
import sys
! hostname
! whoami
print(sys.executable) # OK if generates an exception message on Windows machines
# tested ok MacOS, arm linux, x86-64 linux, Windows 10VM, Windows Server
```

```
atomickitty.aws
engr1330content
/opt/jupyterhub/bin/python3
```

---

## 0.8 Question 2 (5 pts):

The script below is intended to print out all the leap years from 1952 to 1996 (inclusive). However, the programmer keeps getting an error and has become frustrated with it! Please have a look at the code, the error(s), and the output and fix it so that it does what the user wants. Then, explain what was wrong with the script and how you have managed to fix it.

[2]:

```
for years in range(1952,1996,4.0):
  print(years)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-2-d7c12f7d99fa> in <module>
----> 1 for years in range(1952,1996,4.0):
      2   print(years)

TypeError: 'float' object cannot be interpreted as an integer
```

### 0.8.1 Deliverables

- Error-free script
- Correct Output (years 1952,1956 …. , 1996)
- Explain the repairs (either direct comment in the code or narrative in markdown cell)

---

## 0.9 Question 3 (10 pts):

The script below is intended to compute the cosine of values from 0 to 10 (inclusive), store them in a separate list called ylist, and ultimately print out a table. However, the programmer keeps getting an error and has become frustrated with it! Please have a look at the code, the error(s), and the output and fix it so that it does what the user wants. Then, explain what was wrong with the script and how you have managed to fix it.

[3]:

```
xlist = ()
ylist = ()

print("     Cosines     ")
print("   x    ","|"," cos(x) ")
print("--------|--------")

for x in range(0,10):
    xlist.append(x*1.0)
    ylist.append(math.cos(x*1.0))
    print("%.3f" % xlist[x], "  |", " %.2f "  % ylist[x])
```

```
     Cosines
   x     |  cos(x)
--------|--------
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
```

```
<ipython-input-3-07d3d368a680> in <module>
      7
      8 for x in range(0,10):
----> 9     xlist.append(x*1.0)
     10     ylist.append(math.cos(x*1.0))
     11     print("%.3f" % xlist[x], "  |", " %.2f "  % ylist[x])

AttributeError: 'tuple' object has no attribute 'append'
```

### 0.9.1 Deliverables

- Error-free script
- Correct Output (a table of x and cos(x)) for x values of (0.000,1.000,...,10.000)
- Explain the repairs (either direct comment in the code or narrative in markdown cell)

---

# 1 Question 4 (10 pts):

Create a function, named "myplotfunc" that takes the following positional arguments as inputs:

- A list of numeric for x
- A list of numeric for y
- A string for the plot title
- A string for the x-axis label
- A string for the y-axis label

and produces (plots) a curve with "red" color, and a '*' shaped marker, and the 'dashdot' linestyle.

Use your "myplotfunc" function to create a plot of:

$$y = -5x^5 + 4sin(x)^4 - 3x^3 + 2cos(x)^2 - x$$

for x raging from -10 to 10 (inclusive) in steps of 0.5.

[4]:
```
# build your script here
```

### 1.0.1 Deliverables

- Error-free script
- Correct Output (a plot of x and y) for x values of (-10.0,-9.5,...,10.0)
  - Correct color for curve
  - Correct line style
  - Correct marker type
  - Axes labels meaningful
  - Plot title meaningful

---

### 1.1 Question 5 (20 pts):

Follow the steps below: 1. STEP 0: install necessary libraries (numpy and pandas) 2. STEP 1: There are 8 digits in your R#. Define a 2x4 array with these 8 digits, name it “Rarray”, and print it

[5]:
```
# step 0 and step 1 go here!
```

3. STEP 2: Find the maximum value of the “Rarray” and its position

[6]:
```
# step 2 goes here!
```

4. STEP 3: Sort the “Rarray” along the rows, store it in a new array named “Rarraysort”, and print the new array out

[7]:
```
# step 3 goes here!
```

5. STEP 4: Define and print a 4x4 array that has the “Rarray” as its two first rows, and “Rarraysort” as its next rows. Name this new array “DoubleRarray”

[8]:
```
# step 4 goes here!
```

6. STEP 5: Slice and print a 4x3 array from the “DoubleRarray” that contains the last three columns of it. Name this new array “SliceRarray”.

[9]:
```
# step 5 goes here!
```

7. STEP 6: Define the “SliceRarray” as a panda dataframe:
   - name it “Rdataframe”,
   - name the rows as “Row A”,“Row B”,“Row C”, and “Row D”
   - name the columns as “Column 1”, “Column 2”, and “Column 3”

[10]:
```
# step 6 goes here!
```

8. STEP 7: Print the first few rows of the “Rdataframe”.

[11]:
```
# step 7 goes here!
```

9. STEP 8: Create a new dataframe object (“R2dataframe”) by adding a column to the “Rdataframe”, name it “Column X” and fill it with “None” values. Then, use the appropriate descriptor function and print the data model (data column count, names, data types) of the “R2dataframe”

[12]:
```
# step 8 goes here!
```

10. STEP 9: Replace the **‘None’** in the “R2dataframe” with 0. Then, print the summary statistics of each numeric column in the data frame.

[13]:
```
# step 9 goes here!
```

11. STEP 10: Define a function based on the equation below:

$$y = x^2 - 5x + 7$$

apply the function to the entire “R2dataframe”, store the results in a new dataframe (“R3dataframe”), and print the results and the summary statistics again.

[14]:
```
# step 10 goes here!
```

12. STEP 11: Print the number of occurrences of each unique value in “Column 3”

[15]:
```
# step 11 goes here!
```

13. STEP 12: Sort the data frame with respect to “Column 1” with a descending order and print it

[16]:
```
# step 12 goes here!
```

14. STEP 13: Write the final format of the “R3dataframe” on a CSV file, named “Rfile.csv”

[17]:
```
# step 13 goes here!
```

15. STEP14: Read the “Rfile.csv” and print its content. ** **Make sure to attach the “Rfile.csv” file to your midterm exam submission.**

[18]:
```
# step 14 goes here!
```

---

## 1.2 Question 6 (10 pts):

Create a class to compute the average grade (out of 10) of the students based on their grades in Quiz1, Quiz2, the Mid-term, Quiz3, and the Final exam.___

| Student Name | Quiz 1 | Quiz 2 | Mid-term | Quiz 3 | Final Exam |
|---|---|---|---|---|---|
| Harry | 8 | 9 | 8 | 10 | 9 |
| Ron | 7 | 8 | 8 | 7 | 9 |
| Hermione | 10 | 10 | 9 | 10 | 10 |
| Draco | 8 | 7 | 9 | 8 | 9 |
| Luna | 9 | 8 | 7 | 6 | 5 |

1. Use docstrings to describe the purpose of the class.
2. Create an object for each student and display the output as shown below.

“Student Name”: **Average Grade**

3. Create and print out a dictionary with the student names as keys and their average grades as data.

[19]:
```
# script goes here
```

### 1.2.1 Deliverables

- Error-free script
- Print each student and average grade
- Print entire dictionary of students and averages

---

## 1.3 Question 7 (10 pts):

Fun with functions:

When it is 8:00 in Lubbock, - It is 9:00 in New York - It is 14:00 in London - It is 15:00 in Cairo - It is 16:00 in Istanbul - It is 19:00 in Hyderabad - It is 22:00 in Tokyo

Write a function that reports the time in New York, London, Cairo, Istanbul, Hyderabad, and Tokyo based on the time in Lubbock. Use a 24-hour time format. Include error trapping that:

1. Issues a message like "Please Enter A Number from 00 to 23" if the first input is numeric but outside the range of [0,23].
2. Takes any numeric input for "Lubbock time" selection , and forces it into an integer.
3. Issues an appropriate message if the user's selection is non-numeric.

Test your function for these times: - 8:00 - 15:00 - 0:00

[20]:

```
# script goes here
```

### 1.3.1 Deliverables

- Error-free script
- Runs that demonstrate:
  - Initial prompt for input
  - Detect non-numeric entry issues message
  - Detect numeric entry, if acceptable perform time conversions
- Runs (3) for the requested times

---

### 1.3.2 Question 8 (20 pts):

Write a pseudo-code or an algorithm for the problem below. Add comments and clearly explain what you want to happen in each step and why. **Describe the algorithm, DO NOT write a python script for this problem – a flowchart, psuedo-code, or detailed algorithm is sufficient**

*The "Treasure Hunt Problem" is from the HackerRank.com avaiable at https://www.hackerrank.com/contests/startatastartup/challenges/treasure-hunt*

#### Tresure Hunt Problem

```
+-------------------------+
¦ 34 ¦ 21 ¦ 32 ¦ 41 ¦ 25  ¦
+----+----+----+----+-----¦
```

```
¦ 14 ¦ 42 ¦ 43 ¦ 14 ¦ 31  ¦
+----+----+----+----+-----¦
¦ 54 ¦ 45 ¦ 52 ¦ 42 ¦ 23  ¦
+----+----+----+----+-----¦
¦ 33 ¦ 15 ¦ 51 ¦ 31 ¦ 35  ¦
+----+----+----+----+-----¦
¦ 21 ¦ 52 ¦ 33 ¦ 13 ¦ 23  ¦
+-------------------------+
```

Do you like treasure hunts? In this problem you are to write a program to explore the above array for a treasure. The values in the array are clues. Each cell contains an integer between 11 and 55; for each value the ten's digit represents the row number and the unit's digit represents the column number of the cell containing the next clue. Starting in the upper left corner (at 1,1), use the clues to guide your search of the array. (The first three clues are 11, 34, 42). The treasure is a cell whose value is the same as its coordinates. Your program must first read in the treasure map data into a 5 by 5 array. Your program should output the cells it visits during its search, and a message indicating where you found the treasure.

[21]:

```
# change this cell to markdown put your algorithm here
# insert figures in markdown as
# ![](figure_file_name.png)
```

---

### 1.4 Bonus Question for extra credit!

Create a VOLUME Function to compute the volume of Cylinders, Spheres, Cones, and Rectangular Boxes. This function should: - First, ask the user about **the shape of the object** of interest using something like: - *"Please choose the shape of the object. Enter 1 for"Cylinder", 2 for"Sphere", 3 for"Cone", or 4 for"Rectangular Box""* - Second, based on user's choice in the previous step, **ask for the right inputs**. - Third, print out an statement with **the input values and the calculated volumes**.

**Include error trapping that:**

1. Issues a message that *"The object should be either a Cylinder, a Sphere, a Cone, or a Rectangular Box. Please Enter A Number from 1,2,3, and 4!"* if the first input is non-numeric.
2. Takes any numeric input for the initial selection , and force it into an integer.
3. Issues an appropriate message if the user's selection is numeric but outside the range of [1,4]
4. Takes any numeric input for the shape characteristics , and force it into a float.
5. Issues an appropriate message if the object characteristics are as non-numerics.

**Test the script for:**

1. Sphere, r=10
2. r=10 , Sphere
3. Rectangular Box, w=5, h=10, l=0.5

- **Volume of a Cylinder = $r^2h$**

- **Volume of a Sphere = 4( r3)/3**
- **Volume of a Cone = ( r$^2$h)/3**
- **Volume of a Rectangular Box = whl**

[22]:
```
# script goes here
```

### 1.4.1 Deliverables

- Error-free script
- Runs that demonstrate:
    - Initial prompt for input
        - Message for incorrect inputs
    - Secondary prompt for input based on shape
        - Message for incorrect inputs
    - Compute and report values
- Runs (3) for the requested shapes

---