CSCI 127: Joy and Beauty of Data
Snowmester 2020
Final Exam

Friday, January 8, 2021
Instructor: Reese Pearsall

Print your name: _____

Print your Net ID: _____

By submitting this exam, I make the following truthful statements:

• I have not received, I have not given, nor will I give or receive, any assistance to another student taking this exam, nor have I discussed this exam with past students of this course.
• I will not use any non-instructor approved materials to assist me on this exam.
• I will not plagiarize someone else's work and turn it in as my own.
• I understand that acts of academic dishonesty may be penalized to the full extent allowed by the Montana State University Student Code of Conduct, including receiving a failing grade for the exam and/or course. I recognize that I am responsible for understanding the provisions of the Montana State University Student Code of Conduct as they relate to this academic exercise.

Submission Instructions

This exam will be due to D2L by 11:59 PM on Friday, January 8th, 2021. This is a **hard** deadline. Late submissions will receive an automatic 0 for this exam.

The first four pages (**This page and question 1**) will be submitted to the appropriate D2L dropbox as a PDF

- You may print out the exam, fill out the first four pages by hand, scan it/take a picture, and submit as a PDF.
- Or you may fill out the first four pages virtually (i.e. in Word, PDF editor, etc) and save your answers as a PDF.

Problems 2, 3, 4, 5 will be submitted to a separate D2L dropbox. You will develop a python solution for each of those problems. You may have one single .py file for your all your answers, or you may have a separate .py file for each of those problems.

Make sure you name your files accordingly and place your name at the top of each of your .py file in a comment.

You may use notes, lecture slides, in class examples, previous labs/programs, the textbook, lecture recordings, and your computer on this exam.

You CAN NOT use search engines to access external resources (Youtube, Stack Overflow, W3Schools, etc), GroupMe, or other students (previous or current). If you need clarification on a certain question, then you should ask Reese. Reese will make a public announcement to clarify to the entire class if needed.

Be sure to attempt every problem. Even if you cannot fully solve one of the programming problems, you should try to get as much done as you can. I will give out as much partial credit as I can.
Read each question carefully and make sure that you answer everything asked for.

Question 1 (Multiple Choice) (20 pts)

1.1) Match the following Python statements with the correct description

____ Creating a Dictionary	A. `hello = [ ]`
____ Creating a List	B. `hi = hello()`
____ Creating an Object	C. `hi = Hello()`
____ Calling a function	D. `hello = { }`

1.2) What will be the output of the python program given below?

```
def func1(x,y):
     x += 2

def main():
     x = 3
     y = 1
     func1(x,y)
     print(x)

 main()
```

a. 2
b. 3
c. 5
d The program will result in an error

1.3) What will be the output of the python program given below?

```
class Car():
   def __init__(self,make,model,color):
        self.make = make
        self.model = model
        self.color = color
   def str(self):
        answer = ""
        answer += self.color + " " + self.make + " " + self.model
        return answer

car1 = Car("Dodge","Charger","Black")
print(car1)
```

a. Black Dodge Charger
b. Dodge Charger Black
c. It will print out the location of the object in memory
d. The program will result in an error

1.4) What are the two requirements for a recursive function? Write in your answer

1.
2.

1.5) The following code will result in an error. In your own words, **briefly** explain (1) why the error is happening and (2) how you could modify the code to eliminate the error.

```
import numpy as np
a = np.array([1,2,3,4,5])
a = a.reshape(2,2)
print(a)
```

Write your answer in this box

1.6) True/False: `print(names[0])` will always print out the first pair of a dictionary called `names`

a. True
b. False

1.7) True/False**:** You can only have one kind of data type in a numpy array

a. True
b. False

1.8) What is the Python method name for the constructor in a class?

a. `self`
b. `__str__`
c. `__init__`
d. `class`

1.9) What will be the output of the python program given below? *If you think the program will result in an error, please **briefly** describe why.*

```
d = {}
d.append("Hello","World")

print(d["Hello"])
```

a. World
b. Hello
c. Hello World
d. The program will result in an error

If you answered **d**, please briefly describe why you think the program results in an error

Write your answer in this box

1.10) Free question (There is no wrong answer) 😊

What was your favorite lab or program in the class?

Write your answer in this box

Take a deep breath…

I believe in you!

Question 2 (20 pts)

Consider a dictionary that contains information about an airport and the other airports it can fly to. The keys of the dictionary (string) represent a single airport, and the values (list of strings) are the other airports that it has flights to

For example, consider the following dictionary:

```
airports = { "Los Angeles":["Boston","New York","Denver","Salt Lake City"],
             "Billings":["Denver","Seattle","Salt Lake City"],
             "Bozeman":["Denver","Billings"],
             "Denver":["Los Angeles","Bozeman","New York","Boston","Atlanta"],
             "Atlanta":["Dallas"]
            }
```

This would mean that Los Angeles has flights to Boston, New York, Denver, and Salt Lake City. Billings has flights to Denver, Seattle, and Salt Lake City. Bozeman has flights to Denver and Billings, and so on and so forth

A **connecting flight** is a pitstop between the starting airport and the destination. For example, if someone in Billings wanted to fly to Los Angeles, they would have to make a pit stop in Denver, because Billings does not have any flights directly to Los Angeles, but Denver does.
(Billings -> Denver -> Los Angeles)

Using the provided code below, write a function that takes in an input starting airport (Ex. Billings) and an input desired destination airport (Ex. Los Angeles). Your function should calculate the number of flights it would take to go from the starting airport to the destination airport. If there is not a direct flight to go from the start to destination, then your function should check if you can go from the starting airport to destination airport with **one** connecting flight. If the starting airport cannot travel to destination airport directly and cannot travel to the destination airport through **one** connecting flight, then a different message should be printed out (see sample outputs)

```
## Your function should above this line. Do not change anything below this line
airports = { "Los Angeles":["Boston","New York","Denver","Salt Lake City"],
             "Billings":["Denver","Seattle","Salt Lake City"],
             "Bozeman":["Denver","Billings"],
             "Denver":["Los Angeles","Bozeman","New York","Boston","Atlanta"],
             "Atlanta":["Dallas"] }

start = input("What is your starting airport?")
destination = input("What airport is your destination?")

calculate_number_of_flights(start,destination,airports)
```

Below are 4 different example outputs that may be produced

```
What is your starting airport?Billings
What airport is your destination?Denver
To go from Billings to Denver will take 1 flights
```

(Billings has a direct flight to Denver, so it only takes 1 flight)

```
What is your starting airport?Bozeman
What airport is your destination?Boston
To go from Bozeman to Boston will take 2 flights
```

(Bozeman does not have a direct flight to Denver, but it can reach Boston through a connecting flight in Denver (Bozeman -> Denver -> Boston), so it takes 2 flights)

```
What is your starting airport?Bozeman
What airport is your destination?Dallas
Bozeman does not go to Dallas or it would take more than two flights
```

(Bozeman does not have a direct flight to Dallas, and cannot reach Dallas through one connecting flight, so the following message should get printed out)

```
What is your starting airport?Bozeman
What airport is your destination?Chicago
Bozeman does not go to Chicago or it would take more than two flights
```

(Bozeman does not have a direct flight to Chicago, and cannot reach Chicago through one connecting flight, so the following message should get printed out)

*The only possible outcomes are **1** flight, **2** flights, or **not possible**. If you can't solve the entire problem, remember to try to get as much done as you can! If you cant get the connecting flights to work, at least try to get your function to correctly identify the cases where it only takes 1 flight.*

Question 3 (20 pts)

Using the given code below, supply the missing code using object-oriented programming. Each animal has a name (Ex. Panda), the continent where it lives (Ex. Asia) and the color of the animal (Ex. Black and White). The `find_by_location()` method takes in a user input location (Ex. Asia) and prints out the number of animals in the zoo that are from that location. Numpy should **not** be used for this question.

When your program is run, these are two example possible outputs

```
Welcome to the Montana Zoo
These are the animals we have:
*Black and White Panda from Asia
*Pink Flamingo from South America
*Orange Tiger from Asia
*Gray Rhino from Africa

Enter a continent: asia
There are 2 animals in the Montana Zoo from asia

------------------------------

Welcome to the Montana Zoo
These are the animals we have:
*Black and White Panda from Asia
*Pink Flamingo from South America
*Orange Tiger from Asia
*Gray Rhino from Africa

Enter a continent: North America
There are 0 animals in the Montana Zoo from North America
```

```
## Your code should go above this line. Do not change anything below

zoo = Zoo("Montana Zoo")

animal1 = Animal("Panda","Asia","Black and White")
zoo.add_animal(animal1)

animal2 = Animal("Flamingo","South America","Pink")
zoo.add_animal(animal2)

animal3 = Animal("Tiger","Asia","Orange")
zoo.add_animal(animal3)

animal4 = Animal("Rhino","Africa","Gray")
zoo.add_animal(animal4)

print(zoo)

location = input("Enter a continent: ")
zoo.find_by_location(location)
```

Question 4 (20 pts)

Fill in the missing code so that the output graphically displays (print) the frequencies contained in some input integer array. The number at each index will represent the number of stars (*) that should be printed out for that specific index. For example, if index 1 contained the value 4, your program should print out 4 stars for 1 (see desired output). You can assume that the elements of the array will be non-negative and will always contain a valid value.

```
import numpy as np

## Your code will go here. Do not change anything below this comment

def main():

    array = np.array([2,4,7,0,1])
    a = ArrayGraphic(array)

    a.displayGraphic()

main()
```

Desired output:

```
0:**
1:****
2:*******
3:
4:*
```

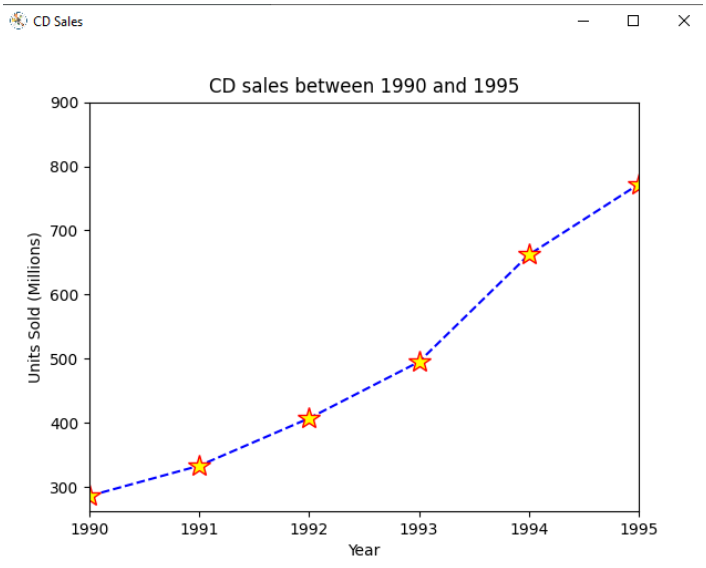
(Index 0 has a 2, so two stars were printed out. Index 1 has a 4, so four stars were printed out. Index 2 has a 7 so seven stars were printed out …)

Question 5 (20 pts)

Consider the following data of number of CDs sold between the years 1990 and 1995

Year	Units Sold (Millions)
1990	286
1991	333
1992	407
1993	495
1994	662
1995	772

Using this data, write a Python program that generates a line chart that looks exactly like this:



The chart has a **blue dashed** line. The markers are **yellow stars**. The edge of the markers are **red**

You are not given any starting code. You will develop this program from scratch. You may find some the examples we did in class helpful for this question. You are allowed to import **Matplotlib** and **Numpy**, but nothing else.

Hint: Read the documentation for matplotlib.pyplot.plot