# CSCI 127: Joy and Beauty of Data
# Summer 2021
# **Practice Final Exam**

Thursday, June 17, 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 8:00 AM on Friday, June 18th, 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 may also use the official documentation for numpy, matplotlib, and pandas**

You CAN NOT use search engines to access external resources (Youtube, Stack Overflow, W3Schools, etc), Discord, or other students (previous or current).

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) Which of the following data types **can** be a key in a Python dictionary?

a. string
b. integer
c. object
d. all of the above

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

```
my_dict = {"reese":”pearsall”, "sam":"smith", "john":"paxton"

for each,value in my_dict.items():

    if each == “reese”:
        print(value)
    else:
        print(each)
```

a. reese
   john
   sam
b. reese
   pearsall
   john
   sam
c. pearsall
   sam
   john
d . pearsall
   smith
   paxton

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

```
class Dog():
    def __init__(self,name,breed):
        self.name = name
        self.breed = breed

    def getName(self):
        return self.breed
dog1 = Dog(“Fido”,”Pug”)
print(dog1.getName())
```

a. Fido
b. Pug
c. self
d. It will print out the location of the object in memory

1.4) Suppose you want to open a file names “scores.txt” for reading. What is the correct python statement?

```
a. file = open(scores.txt,r)
b. f = open("scores.txt","r")
c. my_file = open("scores.txt")
d. input_file = File("scores.txt","r")
```

1.5) 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

1.6) True/False: An object can be an input parameter to a function

a. True
b. False

1.7)True/False**:** Numpy arrays are a fixed size which cannot be changed after initialization

a. True
b. False

1.8) The following code will result in errors. Briefly explain why and how to fix it

```
class Bike():
   def __init__(self,color,wheels):
        self.color = color
        self.wheels = wheels

   def get_color():
        return color

   def get_wheels():
        return wheels

my_bike = Bike("red",2)
print(my_bike.get_color())
```

Write your answer in this box

| |
|---|
| |

1.9) Which of the following Python lines would create an array of size 10 that is filled with zeros:

a. `a = np.zeros(10,dtype=int)`
b. `a = np.array([0,0,0,0,0,0,0,0,0,0])`
c. `a = np.zeros(10)`
d. All of the above are correct

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

```
food = {"Popcorn":5.99, "Soda":1.99, "Nachos":3.88}

food["Soda"] += 2

print(food["Soda"])
```

a. 1.99
b. 3.99
c. 2
d. The following program will result in an error

## Question 2 (20 pts) Dictionaries

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) OOP**

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) OOP + Inheritance**

Fill in the missing code below using object-oriented programming. Consider three different classes that represent different grades at an elementary school.

- A **first-grade class** will have a teacher, a teaching assistant, room number, and class roster.
- A **third-grade class** will have a teacher, a book that students are required to read, a room number, and class roster.
- A **fifth-grade class** will have a teacher, a field trip location, a room number, and class roster

You will implement **inheritance** in this problem. Each of those classes described above should inherit from a parent class called **`Classroom`** which will contain the shared attributes/field and methods. When your program is run, it should produce the following output

```
Mrs. Richmond's class
Room 113
1. Casey
2. Kylee
3. Jonah
4. Sarah

Mrs. Severson's class
Room 140
1. Jacob
2. Haley
3. Noah
4. Angel

Ms. Hill's class
Room 132
1. Bailey
2. Matt
3. Connor
4. Alyssa
```

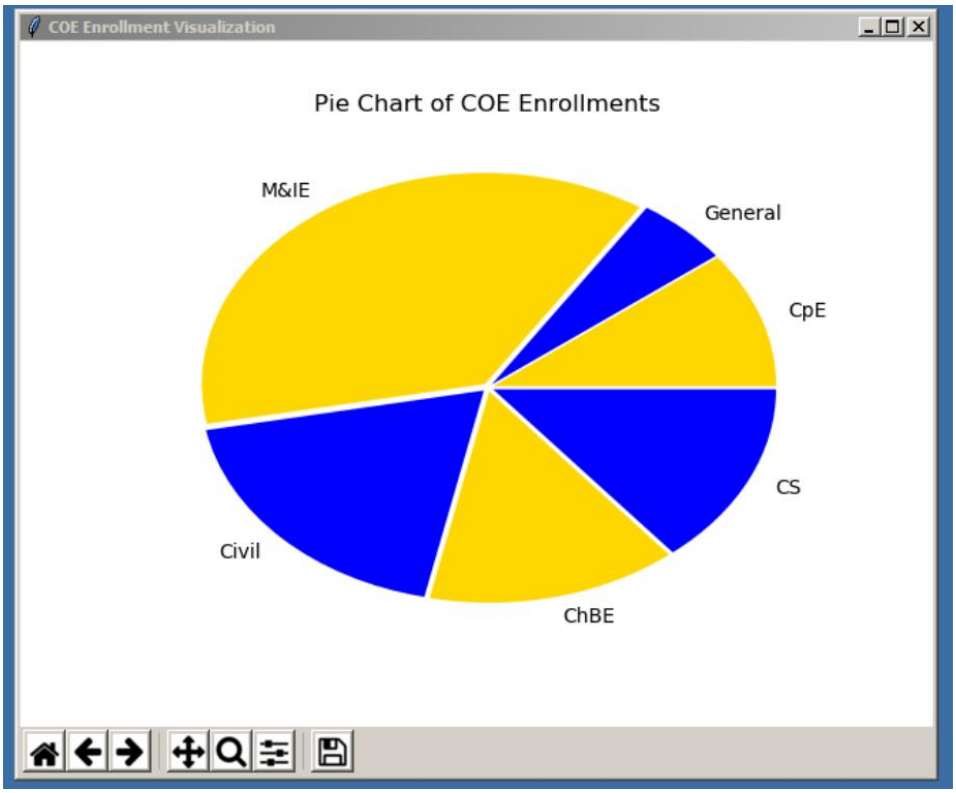
```
import numpy as np

# Your code and four classes should go above this comment. Do not change anything below
def main():

    first_graders = np.array(["Casey","Kylee","Jonah","Sarah"])
    first_grade = FirstGrade("Mrs. Richmond","Ms. Stacey","Room 113",first_graders)
    print(first_grade)

    third_graders = np.array(["Jacob","Haley","Noah","Angel"])
    third_grade = ThirdGrade("Mrs. Severson","Sideways Stories from Wayside School","Room 140",third_graders)
    print(third_grade)

    fifth_graders = np.array(["Bailey","Matt","Connor","Alyssa"])
    fifth_grade = FifthGrade("Ms. Hill","Pictograph Caves","Room 132",fifth_graders)
    print(fifth_grade)


main()
```

Not using inheritance will lead to a significant loss of points. **Hint**: Reference the `employee_oop.py` example we did in class.

**Question 5 (20 pts) Matplotlib**

Complete the Python program below such that the following visualization is produced. The pie wedge color for CS, Civil and General is blue. The other pie wedge color is gold. Notice that there is a small gap between each pie wedge.

Hint: Read the documentation for matplotlib.pyplot.pie




```
import numpy as np # Do not import anything else
import matplotlib.pyplot as plt

units = ["CS", "ChBE", "Civil", "M&IE", "General", "CpE"]
enrollments = [552, 563, 731, 1463, 210, 410] # See question 1 for a description

# Write the missing statements below this comment
```