2143 - OOP
Spring 2020
Final Exam
May 11, 2020

Name: _____

READ THESE INSTRUCTIONS
• Create a digital document that has zero handwriting on it. • Your presentation and thoroughness of answers is part of your grade. I make every effort to create clear and understandable questions. You should do the same with your answers. • Questions should be answered in order and clearly marked. • Your name should be on each page, in the heading if possible. • Turn your final in by uploading to D2L. Link should be obvious on D2L course main page. • **Some of the test questions will deal with topics that we didn't to cover in class, but I still feel you should know about (thanks Covid 19). That's why I'm giving you three days to take the exam!**
Failure to comply will result in loss of letter grade

Grade Table (don't write on it)

Question	Points	Score
1	20	
2	30	
3	30	
4	20	
5	30	
6	20	
7	20	
8	50	
9	25	
10	50	
Total:	295	

1. (20 points) What is the main difference between a class and an object?.

---

2. (30 points) Define the following:

   - **(10 points)** Polymorphism
   - **(10 points)** Encapsulation
   - **(10 points)** Abstraction

Note:

Make sure you give examples as well as compare and contrast when defining the above items.

---

3. (30 points) Discuss constructors for objects. In your discussion hit on these points:

   - **(10 points)** Are they always necessary?
   - **(10 points)** What is a copy constructor?
   - **(10 points)** Deep vs Shallow copy?

---

4. (20 points) What is the difference between an abstract class and an interface?.

Note:

You should include in your discussion:

- Virtual Functions
- Pure Virtual Functions

---

5. (30 points) What are the differences between:

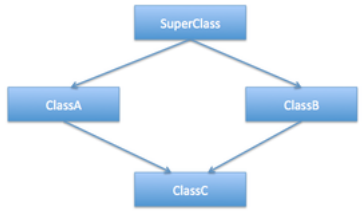
   - **(10 points)** Public
   - **(10 points)** Private
   - **(10 points)** Protected

Note:

Again, use examples if possible. Make sure you define each item individually as well.

---

6. (20 points) What is the diamond problem?



Note:

This is a question about multiple inheritance and its potential problems. Use examples when possible, but explain thoroughly.

---

7. (20 points) Discuss Early and Late binding.

Note:

This is another topic we didn't touch on but is important. You should have these keywords present when answering this question: **static, dynamic, virtual, abstract, interface**. I can't say it enough, try to use examples in your discussion.

---

8. (50 points) What is a design pattern? Describe the following patterns and give examples of when to use:

- **(10 points)** Facade
- **(10 points)** Singleton
- **(10 points)** Factory
- **(10 points)** Observer

---

9. (25 points) Write a class called NumObjects that counts the number of objects in existance. You should assume that each object will be created and destroyed before your program ends. So your count should be equal to the number of existing objects.

Note:

You need the **static** keyword.

---

10. (50 points) Writing a **Grayscale Class**

So what is gray scale? Its where you take the 3 individual parts of a color and using those values you calculate a single value that will be assigned to each of the 3 components, making it some shade of gray.

For example here is red: $(0, 255, 0)$ and here is the gray scale equivalent: $(85, 85, 85)$ (using the average method from below).

Your GrayScaler class is serious about its grayscalin‘ powers and has four methods to turn a color into its monochromatic equivalent:

- Lightness
- Average
- Luminosity
- Custom

Lightness

- The lightness method averages the most prominent and least prominent colors:
- $(max(R, G, B) + min(R, G, B))/2$.

Average

- The average method simply averages the values:
- $(R + G + B)/3$.

Luminosity

- This method also averages the values, but it forms a weighted average to account for human perception. We're more sensitive to green than other colors, so green is weighted most heavily:
- $0.21 * R + 0.72 * G + 0.07 * B$

Custom

- This method allows the user to pass in three floats to act as the weights in your formula:
- $w1 * R + w2 * G + w3 * B$

Here is some example usage to help you determine how to design your class:

```
c1 = Color(255,0,0)
grayish = GrayScaler(c1)
gray1 = grayish.Average()
# gray1 now contains (85,85,85)
gray2 = grayish.Custom(.25,.48,.27)
# gray2 now contains (64,64,64)

grayish2 = GrayScaler() # defaults to black in the class if no color provided
grayish2.SetColor(255,192,203)
gray3 = grayish2.Luminosity()
# gray3 now contains (206,206,206)
```

Write your class in C++ or Python. Its up to you. The example code below is python (obviously) but you can write your implementation in whichever. You should have a full working class that either extends a color class, or is composed of colors.

```
"""
@Description: Gets an RGB color represented as a tuple, and converts it to a gray
   scale equivalent based on chosen method.
@Methods:
    Lightness - as described above
    Average - as described above
    Luminosity - as described above
    Custom - as described above
    SetColor - Lets user change the color originally passed in.
"""
class GrayScaler(Object):

    '''
    Your implementation
    '''


```