Module 8:

Constructors and Other Tools

COP2274

In-class Assignments

UF UNIVERSITY of FLORIDA

M8A Class called ProduceBox
(default constructor *vs.* custom constructor)

1. Write **a class** called **ProduceBox** with *private member variables* for the number of apples, bananas, kiwi, carrots, and broccoli.
2. Write ***a default constructor*** with an initialization section that initializes all of *the member variables* to 0.
3. Write ***a custom constructor*** with five parameters that initializes each *member variable* with its corresponding parameter.
4. Write *a member function* called *print()* to print the quantities of produce in the box as shown in the test cases.

M8A Class called ProduceBox (default constructor vs. custom constructor)

5. In your main(), declare a **ProduceBox** object that is initialized by ***the default constructor,*** and call *print()* on the object. Then, declare another **ProduceBox** object that is initialized by ***the custom constructor*** with the values shown in the test cases, and call *print()* on the object.

Test cases

```
Box of Produce
--------------
Apples: 0
Bananas: 0
Kiwi: 0
Carrots: 0
Broccoli: 0

Box of Produce
--------------
Apples: 6
Bananas: 7
Kiwi: 2
Carrots: 8
Broccoli: 1
```

M8B Class called Discount
(explicit call with anonymous object)

1. Write **a class** called **Discount** with *private member variables for the discount percentage and item price.*
2. Write ***a default constructor*** with an initialization section that initializes all of the member variables to 0.
3. Write ***a custom constructor*** with two parameters that initializes each *member variable* with its corresponding parameter after validating the parameters. The discount percentage must be an integer between 0-100, and the item price must be a double that is not less than 0. If an invalid parameter is passed, set the corresponding member variable to 0 and display an error message as shown in the test cases.

M8B Class called Discount
(explicit call with anonymous object)

4. Write a member function called *calFinalPrice()* that returns the final price after discount.
5. Write ONLY accessors (NO mutators) for each *member variable*. Make all of the accessors as a ***constant*** function.
6. In your main(), declare a **Discount** object that is initialized by the **default constructor**, prompt the user inputs for the discount percentage (int) and the initial price (double), and then reinitialize the member variables of the **Discount** object by assigning an anonymous object with the ***custom constructor*** to the **Discount** object. You can pass the user inputs as the arguments of the ***custom constructor***.

M8B Class called Discount (explicit call with anonymous object)

7. Finally, print the final price to 2 decimal places as shown in the test cases.

Note:

- *Don't forget to add a question for repeating your program.*

Test cases

```
Welcome to the discount calculator!
Enter discount percentage (integer): 24
Enter initial price: 34.99
Here is the final price of your item: $26.59
Would you like to try again? (y/n) y
Enter discount percentage (integer): 101
Enter initial price: 24.50
Invalid discount! Setting discount as 0...
Here is the final price of your item: $24.50
Would you like to try again? (y/n) y
Enter discount percentage (integer): 101
Enter initial price: -1
Invalid Item price! Setting price as 0...
Invalid discount! Setting discount as 0...
Here is the final price of your item: $0.00
Would you like to try again? (y/n) y
Enter discount percentage (integer): 24
Enter initial price: -1
Invalid Item price! Setting price as 0...
Here is the final price of your item: $0.00
Would you like to try again? (y/n) n
Thanks for using the discount calculator! Goodbye!
```

M8C Class called System (static members)

1. Write **a class** called **System** with *private member variables for CLK frequency (int), hard drive size in GB (int), and number of transistors (double).*
2. Write ***a default constructor*** that initializes the frequency as $2 \times 10^9\ Hz$, the hard drive size as 500 GB, and the number of transistors as 1e10. Define ***the default constructor*** with an initialization section.
3. Write ***a custom constructor*** with three parameters that initializes each member variable with its corresponding parameter. Also, define ***the custom constructor*** with an initialization section.

M8C Class called System (static members)

4. Write an accessor and a mutator for each *member variable*. Make all of the accessors as a constant function.
5. Write a constant *member function* called *GetInfo()* to display the system information as shown in the test case.
6. Create a *static member variable* called networkSize that gets incremented by 1 each time a computer is created (int).
7. Create a *static member function* called *GetNetworkSize()* that returns the value of the *static member variable*, networkSize.

M8C Class called System (static members)

8. In your main(), test your **System** class according to the test cases.

Test cases

```
Created a default System object
System CLK: 2000000000 Hz
Hard Drive Size: 500 GB
Transistor Count: 1e+10
Network Size: 1

Created a custom System object
System CLK: 1400000000 Hz
Hard Drive Size: 250 GB
Transistor Count: 3e+09
Network Size: 2

After upgrading the second System:
System CLK: 1400000000 Hz
Hard Drive Size: 1000 GB
Transistor Count: 3e+09
Network Size: 2
```