

COMP 53: Object Orientation Lab, part 3

Instructions: In this lab, we are going to review operator overloading and static data/functions of classes in object-oriented programming.

- Get into groups of **at most two people** to accomplish this lab.
- At the top of your source code files list the group members as a comment.
- Each member of the group must individually submit the lab in Canvas.
- This lab includes **28 points** in aggregate. The details are given in the following.

1 City and CoastalCity

Extend `coastalcity.h` from the previous lab as follows:

1. Overload operator `*` (multiplication) where the left-hand side (LHS) operand is a coastal city, and right-hand side (RHS) operand is an integer. The returned result is a coastal city with the same name, population and water name as the LHS operand, but the number of beaches is multiplied by the RHS operand (**4 points**).
 2. Overload operator `+` (addition) where both LHS and RHS operands are coastal cities. The returned result is a coastal city where
 - the name is the appendage of the names of operands,
 - the population is the summation of populations of operands,
 - the water name is the appendage of the water names of operands, and
 - the number of beaches is the summation of operands' number of beaches.
- Note:* You can use function `str1.append(str2)` to append two strings `str1` and `str2` (**6 points**).
3. Add unsigned integer `coastalCityCount` as a static data that is supposed to count the number of created `CoastalCity` objects. Make this data private (**2 points**).
 4. Revise the constructor of `CoastalCity` class appropriately so that the static data component from previous step gets updated accordingly (**2 points**).
 5. Define a getter function `getCoastalCityCount()` that returns the current value of `coastalCityCount`. This function must be static (**2 points**).

2 Function main

Include `city.h` (from previous lab) and `coastalcity.h` in `main.cpp`.

1. Initialize `coastalCityCount` to zero in the global scope (**2 points**).
2. The `main` function does the following step by step:
 - (a) Create two coastal cities with the following details (**2 points**):
 - San Diego, population: 1.5 million, water name: Pacific ocean, and number of beaches: 5, and
 - Miami, population: 500000, water name: Atlantic ocean, and number of beaches: 8.
 - (b) Multiply the first coastal city to 5 (using the overloaded `*`) and assign it to the same coastal city (**2 points**).

- (c) Print the information of the first coastal city using `printInfo()` (**1 points**).
- (d) Add first and second coastal cities together (using the overloaded `+`), and assign the result to a new (third) coastal city (**2 points**).
- (e) Print the information of the third coastal city using `printInfo()` (**1 points**).
- (f) Print the number of created coastal cities by calling `getCoastalCityCount()` (**2 points**).

The output of the program may look like the following:

```
Name: San Diego
Population: 1500000
Water: Pacific Ocean
No. of Beaches: 25
```

```
Name: San DiegoMiami
Population: 2000000
Water: Pacific OceanAtlantic Ocean
No. of Beaches: 33
```

4