COMP 53: Object Orientation Lab, part 1

Instructions: In this lab, we are going to review inheritance 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 **18 points** in aggregate. The details are given in the following.

1 Class `CoastalCity`

First, create a header file `city.h` that defines class `City` as follows.

```
#ifndef CITY_H
#define CITY_H

#include<string>
#include<iostream>
using namespace std;

class City {
        public:
                void setName(string name) {this -> name = name;}
                void setPopulation(unsigned int population) {
                        this -> population = population;
                }
                string getName() const {return this-> name;}
                unsigned int getPopulation() const {return this -> population;}
                void printInfo() const {
                        cout<<"Name: "<<getName()<<endl;
                        cout<<"Population: "<<getPopulation()<<endl;
                }

        private:
                string name;
                unsigned int population;
};

#endif
```

In the following, you are going to define `CoastalCity` class that inherits class `City`. Put all your definitions in a separate header file `coastalcity.h` where all functions are inlined (similar to above).

1. Class `CoastalCity` is a derived class that inherits from class `City` ***(2 points)***.
2. `CoastalCity` has two additional data components that are hidden from the class user ***(2 points)***:
   - `waterName`, a string that stores the body of the water adjacent to the coastal city, and
   - `beachNum`, an integer that stores the number of beaches that the coastal city has.
3. Define the default constructor for the class, which does the following:
   - assigns `N/A` to `name`,
   - assigns `0`, to `population`,

- assigns `N/A` to `waterName`, and
- assigns `0`, to `beachNum`.

Note that in the default constructor you are supposed to directly access the data components (rather than calling setters). In this regard, you need to modify class `City` in a way that `name` and `population` are accessible to the subclasses of `City`. However, `name` and `population` should not be accessible to any other function. That is, only the member functions of `City` and its subclasses (e.g., `CoastalCity`) can access them ***(3 points)***.

4. Define the setters and getters for `waterName` and `beachNum` as usual ***(4 points)***. Note that you do not need to define setters and getters for `name` and `population`, as they are inherited from `City`.
5. Override function `printInfo()`. To this end, invoke `City`'s `printInfo()`. This handles printing `name` and `population`. Next, print the water name and number of beaches ***(3 points)***.

2 `Main` function

Define `main` function in `main.cpp` that does the following step by step ***(4 points)***.

1. Creates a coastal city and prints its information, by calling `printInfo()`.
2. Set its name to `San Francisco`.
3. Set its population to `900000`.
4. Set its body of water name to `SF Bay`.
5. Set its number of beaches to `10`.
6. Print its information by calling `printInfo()`.

The output of the program may look like the following:

```
Name: N/A
Population: 0
Water: N/A
No. of Beaches: 0
Name: San Francisco
Population: 900000
Water: SF Bay
No. of Beaches: 10
```