COMP 53: Object Orientation Lab, part 2

Instructions: In this lab, we are going to review compile-time vs. runtime polymorphism 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 **14 points** in aggregate. The details are given in the following.

1 `City` and `CoastalCity`

Include `city.h` and `coastalcity.h` header files from the previous lab in `main.cpp`. The `main` function does the following step by step:

1. Create two pointers to `City` objects and nullify them ***(1 points)***.
2. Create two pointers to `CoastalCity` objects and nullify them ***(1 points)***.
3. Create a vector of pointers to `City` objects ***(1 points)***.
4. Initialize the first pointer to `City` object (using `new`), set the name to `Denver`, and the population to `750000` ***(1 points)***.
5. Initialize the second pointer to `City` object (using `new`), set the name to `Reno`, and the population to `250000` ***(1 points)***.
6. Initialize the first pointer to `CoastalCity` object (using `new`), set the name to `San Diego`, the population to `250000`, the water name to `Pacific Ocean`, and number of beaches to `5` ***(2 points)***.
7. Initialize the second pointer to `CoastalCity` object (using `new`), set the name to `Miami`, the population to `500000`, the water name to `Atlantic Ocean`, and number of beaches to `8` ***(2 points)***.
8. Add all four `City` and `CoastalCity` pointers to the already created vector (Step 3) ***(1 points)***.
9. Within a loop traverse the vector and print each city's information (by calling `printInfo()`) ***(2 points)***.

   Compile and run. Since the vector is statically defined as the vector of pointers to `City` objects, due to **compile time polymorphism**, `printInfo()` in the base class (`City`) is invoked for both `City` and `CoastalCity` object pointers. The output of the program may look like the following:

```
Name: Denver
Population: 750000

Name: Reno
Population: 250000

Name: San Diego
Population: 1500000

Name: Miami
Population: 500000
```

10. Change `city.h` in a way that **runtime polymorphism** is enforced when `printInfo()` function is invoked. That is, for pointers of `City` objects `City`'s `printInfo()` is called, whereas for pointers of `CoastalCity` objects `CoastalCity`'s `printInfo()` is called ***(2 points)***. The output of the program may look like the following:

```
Name: Denver
Population: 750000

Name: Reno
Population: 250000

Name: San Diego
Population: 1500000
Water: Pacific Ocean
No. of Beaches: 5

Name: Miami
Population: 500000
Water: Atlantic Ocean
No. of Beaches: 8
```