COMP 53: Lists Lab, part 4

Instructions: In this lab, we are going to review insertion sort on doubly-linked lists.

- 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 **9 points** in aggregate. The details are given in the following.

1 `city.h` and `citynode.h`

Consider `city.h` and `citynode.h` (`CityNode`s with two links: next and previous) as the one from the previous lab.

2 `citylist.h`

Consider `citylist.h` as the one from the previous lab, where you implemented a doubly-linked list of `CityNodes`.

```
#ifndef CITYLIST_H
#define CITYLIST_H

#include<string>
#include "citynode.h"
class CityList {
        public:
                CityList();
                void append(CityNode *cityNode);
                void prepend(CityNode *cityNode);
                void printCityList();
                CityNode *search(string cityName);
                void insert(CityNode *currNode, CityNode *cityNode);
                void remove(CityNode *currNode);
        private:
                CityNode *head;
                CityNode *tail;

};

#endif
```

Class `CityList` implements the doubly-linked list of cities, which keeps track of the first and last elements of the list (through `head` and `tail` pointers, respectively). You have already completed the definition of functions above in previous labs.

1. Define function `void insertionSortByPopulation()` as part of this class that sorts the list of `CityNode` objects in the ascending form, according to the city population. *Hint*: You will end up using `prepend()`, `remove()` and `insert()` as part of the definition of this function ***(5 points)***.

3 `main.cpp`

In `main.cpp` do the following step by step:

1. Globally define array `cityArray[]` consisting of cities with the following details:
   (a) Los Angeles with population of 4340174
   (b) San Diego with population of 1591688
   (c) San Francisco with population of 871421
   (d) Sacramento with population of 505628
   (e) Stockton with the population of 323761
   (f) Redding with the population of 90292
   (g) Las Vegas with the population of 711926
   (h) Reno with the population of 289485
   (i) Portland with the population of 730428
   (j) Seattle with the population of 752180
   (k) Eugene with the population of 221452
2. Globally define a `CityList` named as `cityList` ***(1 points)***.

In `main()` function do the following step by step, using the functions defined above:

(i) Initialize `cityList` according to array `cityArray[]` by appending, using the function defined above ***(1 points)***.

(ii) Print out the entries of `cityList`, using the appropriate function defined as part of `CityList` class ***(1 points)***.

(iii) Do insertion sort on `cityList` according to the the city populations. Next, print out the updated list ***(1 points)***.

The output of the program may look like the following:

```
Initializing cityList with cityArray[] using appending:
Los Angeles: 4340174
San Diego: 1591688
San Francisco: 871421
Sacramento: 505628
Stockton: 323761
Redding: 90292
Las Vegas: 711926
Reno: 289485
Portland: 730428
Seattle: 752180
Eugene: 221452

Insertion sort of cityList by population:
Redding: 90292
Eugene: 221452
Reno: 289485
Stockton: 323761
Sacramento: 505628
Las Vegas: 711926
Portland: 730428
Seattle: 752180
San Francisco: 871421
San Diego: 1591688
Los Angeles: 4340174
```