Winter 2020 CS 32

Project 4 FAQ

1. **What's with the classes StreetMapImpl, PointToPointRouterImpl, etc.?**

   While the technique has other uses, we're using it to have the compiler prevent you from changing the public interfaces to the classes we're having you implement.

   This example is so simple it seems silly, but it illustrates the point: Suppose we gave you an assignment to implement a Stack class with this required interface:

```
class Stack
{
  public:
        Stack();
        bool empty() const;
        void push(int x);
        int pop();  // remove the top int and return it
};
```

   and forbade you from adding any additional public members. Another part of the assignment requires you to use this type to do some task, and you find it would be convenient if Stack had a public top() member you could use given a reference to a const Stack. If we said that you must turn in Stack.h declaring the class (and Stack.cpp implementing it), you might turn in a Stack.h where the class has an additional public member function top() you added, in violation of our prohibition. While there are ways we could automate checking for that violation, they're more complicated than an alternative technique.

   What we would do instead is give you a Stack.h with the interface, and don't have you turn it in. Since you're not turning it in, we won't see any changes you make to it in violation of the requirement; we'll use the original Stack.h for grading. But if you can't modify Stack.h with its definition of class Stack, how do you add private data members of your choosing?

   The answer is that you don't add those private members to the Stack class. We declare exactly one private member, a pointer to a dynamically allocated object of some type that you have complete control over. Your definition of that type goes in a different file; we may as well have that file be named Stack.cpp. You can define that type however you want, and that file will also contain implementations of the Stack member functions that generally simply ask the dynamically allocated object of your type to do the work. While not mandatory, for ease of understanding it's helpful to have your type have a parallel set of member functions to Stack's. Thus, Stack.h would contain

```
class StackImplementation;

class Stack
{
  public:
        Stack();
        ~Stack();
        bool empty() const;
        void push(int x);
        int pop();
  private:
        StackImplementation* m_impl;
};
```

and Stack.cpp could contain

```
#include "Stack.h"

class StackImplementation
{
  public:
        StackImplementation() : m_top(0) {}
        bool empty() const { return m_top == 0; }
        …
  private:
        int m_data[100];
        int m_top;
};

Stack::Stack() { m_impl = new StackImplementation; }
Stack::~Stack() { delete m_impl; }
bool Stack::empty() const { return m_impl->empty(); }
…
```

You would turn in only Stack.cpp. We would also forbid you from using StackImplementation anywhere else in your code. Thus, if for another task in the assignment that required you to use Stack, you wanted to add a top() function, you couldn't: Adding it to Stack.h wouldn't help you, since you won't turn in that file, and adding it to StackImplementation wouldn't help, since we could easily automatically determine that you were using StackImplementation from that other part of your code where it was forbidden.

In the skeleton we give you, provided.h is the home of StreetMap, PointToPointRouter, etc., StreetMap.cpp is the home of StreetMapImpl, PointToPointRouter.cpp is the home of PointToPointRouterImpl, etc. It's in those cpp files that you'll do the real work.

2. **My program seems to run correctly, but runs awfully slowly, and I know it's not because of bad algorithms (O(N^2) instead of O(N), etc.). How can I speed it up?**

   Especially if it's Visual C++, it may be that you're building in the Debug configuration, which inserts code to check for many STL usage errors. These are helpful during debugging, but can slow things down. Follow the instructions in Homework 4, problem 5 to build an optimized version.

3. **In the map data file, must the street segments be listed in an order that forms one contiguous stretch of roadway?**

   No. The segments listed after the street name in the file may be in any order and might not form just one contiguous stretch. For example, consider this hypothetical street (near Paris, judging by the coordinates):

```
Rue Exemple
4
45.000 3.000  45.000 3.001
45.000 3.001  45.001 3.002
45.001 3.003  45.001 3.004
45.001 3.004  45.000 3.005
```

   This describes a street with two discontiguous stretches, one from (45.000,3.000) east to (45.000,3.001) then northeast to (45.001,3.002) and one from (45.001,3.003) east to (45.001,3.004) southeast to (45.000,3.005). But the file might just as well list the segments in a different order or the endpoints of a segment in the opposite order; the same information is conveyed:

```
Rue Exemple
```

```
4
45.000 3.000  45.000 3.001
45.001 3.004  45.001 3.003
45.000 3.001  45.001 3.002
45.001 3.004  45.000 3.005
```

It still specifies two discontiguous stretches, one consisting of two segments connected at (45.000,3.001) and one consisting of two segments connected at (45.001,3.004).

For the most straighforward implementation of *StreetMap* that we can think of, the code works the same for either of these arrangements and results in the same information being stored as a *StreetMap*'s data.

4. **Is it correct to say that we must NOT use containers defined in `<map>`, `<set>`, `<unordered_map>`, or `<unordered_set>` in our implementations of *ExpandableHashMap* and *StreetMap*, but we ARE allowed to use them for *PointToPointRouter*, *DeliveryOptimizer*, and *DeliveryPlanner*, and we ARE allowed to use *vector*, *list*, *stack*, *queue*, and *priority_queue* for any of the five classes?**

   Yes.

5. **Do we know enough from this class to do a complexity analysis of A* or simulated annealing?**

   No. In your report, for the functions for which you implemented A* or simulated annealing, say so and instead of telling us the big-O for that function, briefly describe the data structures you used.