C++: Practical session 10

1 Basic use of vector, list, and map

1.1 Vector

Write a program to take as input a set of positive floating-point numbers from the user, using a negative number to indicate the end of the set, and put them into a `std::vector`. Next, sort this list in ascending order, and output every other element of the sorted `std::vector`, starting with the first.

Sample program output:

```
Please enter a list of positive numbers, ending with a negative one:
2.3
1.2
9.7
2.718182818
4.2857142857
10.9
100.0
-1
Sorted vector, every other element:
1.2
2.718281828
9.7
100
```

You will need to use `a.push_back(myVal);` which appends an element to a container. The `push_front(myVal)` function is only available for certain containers, such as `list`.

You should first make sure that you can sort the vector and output every element before trying to only output every other one. Note that incrementing `myVector.end()` is undefined, so you will have to be careful.

Now alter your program to use a `std::list` instead. Recall that for a list, `sort()` is a member function.

1.2 Telephone directory

Use a `std::map<std::string, std::string>` to store a telephone directory. Your program should read a set of names and telephone numbers from the user and put these into the directory. It should then allow the user to look up people's telephone numbers from the directory.

Recall that accessing a `std::map` with `[key]` always adds the named element. Use `find` instead. Sample program output:

```
Name: Dr Doolittle
Number: 03402 646257
Name: Prof Smith
Number: 05624 814253
Name: Mr Hyde
Number: 06417 639124
```

```
Name: Dr Jekyll
Number: 06417 639124
Name:
Starting look-up.
Name: Prof Smith
The telephone number for Prof Smith is 05624 814253.
```

Note that in order to allow spaces when inputing a string, you should use:

```
std::getline(std::cin, name);
```

which will read a complete line, including spaces, from `std::cin` into `name`.

1.2.1 Extension

Implement a reverse lookup facility (i.e. determine who's calling, given their number). What is the expected complexity of this operation, compared to simple lookup? Note that, given the above input, the program should print two names for one of the numbers.

Use string processing to parse and store each telephone number in a standard format, such as `+44 01223 746627`. Alternatively, use a class or struct to store the telephone number.