Part B - Foundations

Dynamic Memory

Workshop 3 (worth 3% of your final grade)

In this workshop, you are to process an array of objects where the user specifies the number of elements in the array at run-time.

LEARNING OUTCOMES

Upon successful completion of this workshop, you will have demonstrated the abilities

- to allocate and deallocate dynamic memory for an array
- to use standard library functions to accept input data
- to use standard library functions to format output data

SUBMISSION POLICY

The "in-lab" section is to be completed **during your assigned lab section**. It is to be completed and submitted by the end of the workshop. If you do not attend the workshop, you can submit the "in-lab" section along with your "at-home" section (a 20% late deduction will be assessed). The "at-home" portion of the lab is **due the day before your next scheduled workshop**

All your work (all the files you create or modify) must contain your name, Seneca email and student number.

You are responsible for regularly backing up your work.

IN-LAB SUBMISSION: ITEM CLASS (80%)

Design and code a class named **`Weather`** that holds information about weather conditions. Place your class definition in a header file named **`Weather.h`** and your function definitions in an implementation file named **`Weather.cpp`**. Include in your coding, all of the statements necessary for your code to compile under a standard C++ compiler.

> *Note:*
> *Do not forget that all the modules except main() are implemented within the "**sict**" namespace.*

Do not forget to add compilation safeguards to your header files.

The class `Weather` has the following two member variables:

- A 7-characters long C-style string (8 chars, including the NULL), holding the calendar date associated with the weather information. The date of formatted as follows: (Jan/21, Apr/1, …)
- A `double` value, representing the high temperature on the specified date
- A `double` value, representing the low temperature on the specified date

Your design includes the following member functions:

- `void set(const char*, double, double) -` a modifier that stores valid data in a `Weather` object. The first parameter receives the address of a C-style string that holds the calendar date. The second and third parameters receive the low temperature, and high temperatures, respectively.
- `void display() const` - a query that displays the high and low temperature as well as the associated calendar date in the following format:
    - date left justified in a field of 10 using underscore '_' instead of white space
    - high and low temperatures right justified in fields of 6 with 1 decimal place each using underscore '_' instead of white space

[Hint: use the reference example in the notes to find the right manipulators.]

[Hint: `setw(int)` only affects the next item sent to the stream.]

CLIENT MODULE

Complete the following implementation file for the w3 main module. The missing regions of code are highlighted.

```
// OOP244 Workshop 3: Dynamic Memory
// File     w3_in_lab.cpp
// Version 1.0
// Date     ???
// Author   ???
// Description
// This file is used to demonstrate dynamic memory in
// C++ and to process an array of objects of compound
// type where the user specifies the number of
// elements in the array at run-time.
/////////////////////////////////////////////////////
```

```cpp
#include <iostream>
#include "Weather.h"
using namespace std;
using namespace sict;
int main(){
  int i; //  loop counter
  int n; //the count of days worth of weather
  // initialize the weather pointer here


  cout << "Weather Data" << endl
       << "=====================" << endl
       << "Days of Weather: ";
  cin >> n;
  cin.ignore();

  // allocate dynamic memory here


  for (i = 0; i < n; i++){
    char date_description[7];
    double high;
    double low;

    // ... add code to accept the user input
    // for the weather array

  }
  cout << endl
       << "Weather report:" << endl
       << "Date        high  low" << endl
       << "======================" << endl;

  for (i = 0; i < n; i++){
    weather[i].display();
  }

  // deallocate dynamic memory here

  return 0;
}
```

Output Example:

```
Weather Data
=====================
Days of Weather: 3
Enter date: Oct/1
Enter high: 15
Enter low : 10
Enter date: Nov/13
Enter high: 10
Enter low : 1.1
Enter date: Dec/15
Enter high: 5.5
Enter low : -6.5

Weather report:
Date        high  low
======================
Oct/1_______15.0__10.0
Nov/13______10.0___1.1
Dec/15_______5.5__-6.5
```

For submission instructions, see the SUBMISSION section below.

IN-LAB SUBMISSION

To test and demonstrate execution of your program use the same data as the output example above.

If not on matrix already, upload your **`Weather.h`** and **`Weather.cpp`** and **`w3_in_lab.cpp`** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account:

```
     Sections SAA and SBB:
     ~edgardo.arvelaez/submit w3_in_lab <ENTER>
     Section SCC and SDD:
     ~fardad.soleimanloo/submit w3_in_lab <ENTER>
     Section SEE and SFF:
     ~eden.burton/submit w3_in_lab <ENTER>
```

and follow the instructions.

AT-HOME SUBMISSION: SEARCH (20%)

After completing the solution described, copy w3_in_lab.cpp to w3_at_home.cpp and upgrade the definition of your class to include two more queries:

- **`const char* date() const,`** this query returns the date of a **`Weather`** object.
- **`double low() const,`** this query returns the low temperature of a **`Weather`** object.

Also, add the following global function to your client module, **`w3_at_home.cpp`**.

-
  ```
  double findLow(const char* date, const Weather *data,
                 int dataSize);
  ```

This function takes the following three arguments: The first parameter receives the address of a C-style string that represents the date of a Weather object. The second parameter receives the address of the array made in part one of the workshop. The third argument receives the size of the array made in part one of the workshop. This function searches for a date associated with a Weather object in the array and returns the low temperature of that day. If the date is not found in the data, return 0.0.

The main program that searches for the date in the array of Weather objects contains the following code. The output from this program should exactly look like the following output example.

```
// OOP244 Workshop 3: Dynamic Memory
// File    w3_at_home.cpp
// Version 1.0
// Date
// Author
// Description
///////////////////////////////////////////////////////////


#include <iostream>
#include <cstring>
#include "Weather.h"
using namespace std;
using namespace sict;
double findLow(const char* date, const Weather *data, int dataSize);
```

```
int main(){
  int i; //  loop counter
  int n; //the count of days worth of weather
  // create the weather pointer here

  cout << "Weather Data\n";
  cout << "=====================" << endl;
  cout << "Days of Weather: ";
  cin >> n;
  cin.ignore();
  // allocate dynamic memory here

  for (i = 0; i < n; i++){
    char date_description[7];
    double high;
    double low;

    // ... add code to accept user input for
    //weather

  }
  cout << endl
    << "Weather report:" << endl
    << "Date        high  low" << endl
    << "======================" << endl;

  for (i = 0; i < n; i++){
    weather[i].display();
  }

  char query[7];
  cout << endl << "Enter the date you are looking for: ";
  //accept user input for the date to find
  //(in this example stored in char query[7])
  // and display the found low temprature.

  cin.getline(query, 7, '\n');
  double low = findLow(query, weather, n);
  cout << "Low temperature: " << low << endl;
  // deallocate dynamic memory here

  return 0;

}
```

Output Example:

```
Weather Data
=====================
Days of Weather: 3
Enter date: Oct/1
Enter high: 15
Enter low : 10
Enter date: Nov/13
Enter high: 10
Enter low : 1.1
Enter date: Dec/15
Enter high: 5.5
Enter low : -6.5

Weather report:
Date        high  low
======================
Oct/1_______15.0__10.0
Nov/13______10.0___1.1
Dec/15_______5.5__-6.5

Enter the date you are looking for: Nov/13
Low temperature: 1.1
```

AT-HOME SUBMISSION: REFLECTION (20%)

Please provide brief answers to the following questions in a text file named **`reflect.txt.`**

1) What happens to dynamic memory if it is not deallocated?
2) What are the two methods of formatting IO stream data?
3) Why was dynamic memory required in the main() function of w3_in_lab.cpp to build the solution?

SUBMISSION

To test and demonstrate execution of your program use the same data as the output example above.

If not on matrix already, upload your **`Weather.h`**, **`Weather.cpp, reflect.txt`** and **`w3_at_home.cpp`** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account:

```
Sections SAA and SBB:
~edgardo.arvelaez/submit w3_at_home <ENTER>
Section SCC and SDD:
~fardad.soleimanloo/submit w3_at_home <ENTER>
Section SEE and SFF
~eden.burton/submit w3_at_home <ENTER>
```

and follow the instructions.