

Write a complete C++ program, including comments in each function and in the main program, to do the following to classify groups of three daily temperatures in New York City. The program will determine whether the 3 temperatures are valid. If they are, the program will find their average and classify that average into one of six seasons.

You will write a main program and 5 functions: **validtemperatures**, **isvalid**, **classify**, **findavg**, and **whichseason**. Each is described below.

Outline: The main program will read in a group of three data values representing three daily high temperatures in New York City. The main program will print the temperatures. Then it will call a function to determine if these three values are all valid temperatures (valid means in the range from -15 to 106, including both end points— be sure to test both -15 and 106). If the temperatures are all valid, the main program will call a function to classify the the average of the temperatures into a season. The main program will repeat this process for the entire set of data.

The Main Program:

The main program will begin by calling the function **introduction**.

Then, in a loop, the main program will read in and process groups of three integer values until the end of the set of data. For each group of 3 values, the main program will print the 3 temperatures and send all 3 as parameters to the function **validtemperatures** to determine whether all three numbers represent valid temperatures.

If the function **validtemperatures** says all 3 temperatures are valid, the main program will print "the group of temperatures is valid," add to a counter of valid groups and call **classify**. However, if **validtemperatures** says the group is not valid, the main program will print "the group of temperatures is invalid," and add to a counter of invalid groups. In both cases, the main program will also count the total number of groups of temperatures processed.

The main program will continue to read and process groups of temperatures until the entire set of data has been processed. After processing all the groups of temperatures, the main program will print the final values of three counters. Use three separate counters. Do **not** compute one from the others.

Functions:

1. The function **introduction** will not receive any parameters (and will not return an answer). This function will print your name, the assignment number, and a brief description of what the program does. You must include a list of the seasons and the temperature ranges for each season.

2. The function **validtemperatures** will receive three integers as parameters. The function will determine whether or not the three temperatures are valid—each in the range -15 to 106. The function will print each invalid temperature together with a message explaining why it is invalid (print one message if it the temperature is too high, and a different message if the temperature is too low).

The function **validtemperatures** will keep track of how many temperatures in each group are valid (it can use a boolean operation or count). If all temperatures are valid, **validtemperatures** will return **true** (or 1) to the main program; if any temperature is invalid (or if all are), the function will return **false** (or 0).

To determine whether or not the temperature is valid, the function **validtemperatures** will send the temperatures one at a time to a function **isvalid** (that means 3 calls, sending one temperature each time). The function **validtemperatures** will combine the results from the 3 function calls and return a single value to the main program. If all 3 temperatures are valid, **validtemperatures** will return **true**; if any temperature is invalid, **validtemperatures** will return **false**. This function MUST return a Boolean value.

3. The function **isvalid** will receive one integer representing a temperature. If the temperature is not valid, **isvalid** will print it, together with a message explaining why it is invalid—for example, if the function receives -20, it will print "-20 is too small"; if it receives 125, it will print "125 is too large". If the temperature is valid, **isvalid** will not print anything.

The function **isvalid** will ALSO return a value indicating whether the temperature is valid or invalid. If the temperature is valid, **isvalid** will return 1 (or **true**) to **validtemperatures**, indicating that this is a valid temperature. If the temperature is invalid, the function will return 0 (or **false**) to **validtemperatures**.

The function **isvalid** can return either an integer (1 or 0) or a Boolean (**true** or **false**)—your choice. **Advice:** if you are going to add values in **validtemperatures**, use **int**. If you are going to use logical operators, use **bool**.

4. The function **classify** will be called ONLY if **validtemperatures** returns a result indicating that all 3 temperatures are valid. The function **classify** will receive 3 integers and send them to the function **findavg**, which will return to **classify** the average of the three temperatures. Then **classify** will send the average to the function **whatseason** to determine what season this average represents. Then **classify** will return to the main program.

5. The function **findavg** will receive three integers as parameters. It will return the average of the 3 integers, which will be a **double** (be sure to divide by 3.0 to get decimal places!).

6. The function **whatseason** will receive the average temperature (a **double**) as a parameter and use that temperature to determine the season. There are six seasons. The function will print the average temperature and, next to it, the season, according to the following pattern:

If the average temperature is 95 or above, it is roasting season;
if it is from 80 up to 95 (but not including 95), it is summer;
from 65 up to 80 (but not including 80) is spring;
45 up to 65 (but not including 65) is fall;
10 up to 45 (but not including 45) is winter
below 10 is freezin' season.

You must use a cascaded **if-else** in this function; model it on the **last** version of **whichweek** from Program 6. This is the version with one test per season.

IMPORTANT:

You will write two versions of the main program. Do the second ONLY AFTER have approved your first version.

First Version of the main program:

1. Use **cin** for input and a file for output.
2. Use a do-while loop with user-response method to determine the end of the set of data.
3. Use about 5 sets of data: one all valid, one all invalid, and 3 combinations of high, low, and valid.

Second version of the main program:

1. Use a file for input and a file for output.
2. Use a while loop and the reading-to-input-failure method to determine the end of the set of data.
3. Use at least 25 sets of data (see guidelines below).

STYLE:

Each function should have a good comment explaining its role in the program and what parameters it will receive. If the function calls another function, the comment should say that.

DATA and OUTPUT:

You will be judged on the quality of your data. In version 2, organize your data sets in the file so that you can clearly see what you have tested.

Include at least 25 groups of data values. Have some values below -15 and some above 106. Include at least 10 invalid groups, three with just one invalid (in different positions), two groups with two invalid temperatures (in different positions), and two with all three invalid, including one with all too low and one with all too high. For the valid groups (have at least 15-20), make sure the data tests every season in the **whatseason** function. Make sure you test the endpoints of all categories (45 and 65, for example).

OUTPUT: Here is sample output for two groups of temperatures:

the 3 values are: 85 92 104

the set of temperatures is valid

the average is 93.66

the season is summer

1 set was valid

1 set was invalid

there were 2 sets of data

the 3 values are: -30 140 80

30 is too small

140 is too large

the set of temperatures is invalid

Plan of action:

First: write a pseudocode outline of the program. Submit it for feedback if you wish.

Write the main program (version 1) and the functions. Write and test one function at a time. Test your program by entering data from the keyboard (cin): The user-response method is easy to use for testing (but hard to use if you need many data sets).

For version 2, create a list of your data sets in a file using your editor. Organize the data sets to make sure that they cover all the required combinations. Save the file, calling it something like data4.txt.

What to Submit:

Version 1, submit your program file (.cpp) and your output file attached to an email with the subject line **A4-1 1110**.

Version 2, submit your program file (.cpp), your data file, and your output file attached to an email with the subject line **A4-2 1110**.