Name: Otto Wiking Höger

Date: 3/11/2024

Spring 2024

CSCI 1300: Recitation 8

Please make sure to write your name and the date in the top left corner. You may use any course materials to answer the following questions and you may collaborate with others, but the work shown must be your own.

1 Spot The Error

Problem 1.1. The program is supposed to flip the sign of all the numbers in an array. Identify the error(s) in the code below, and write the correct line(s).

```
#include <iostream>
#include <string>
using namespace std;

void flipSign(int numbers[])size)
{
    for (int i = 0; i < 10; i++)
    {                       size
        numbers[i] = -1 * numbers[i];
    }
    return;
}

int main()
{
    const int length = 4;
    int numbers[] = {1, 2, 3, 4};

    cout << "The contents of the array before changing: ";
    for (int i = 0; i < length; i++)
    {
        cout << numbers[i] << " ";
    }
    cout << endl;

    flipSign(numbers);

    cout << "The contents of the array after changing: ";
    for (int i = 0; i < length; i++)
    {
        cout << numbers[i] << " ";
    }

    return 0;
}
```

Name:

Date:

Spring 2024

CSCI 1300: Recitation 8

Problem 1.2. The program below reads a file visitors.txt and prints out the busiest day. Each line in the file has the format:

`dayOfWeek <space> visitor1,visitor2,..,visitorN.`

Identify the error(s) in the code below and write the correct line(s).

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    ofstream my_file("visitors.txt");
    string full_line;
    string days[] = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday"};
    int vis[5] = {0, 0, 0, 0, 0};
    int i = 0;
    int traffic = 0;
    int j = 0;

    while (getline(my_file, full_line))
    {
        for(int i = 0; i < static_cast<int>(full_line.length()); i+=1)
        {
            if(full_line[i] == ' ' && i < static_cast<int>(full_line.length())-1)
            {
                visitors[i]++;
            }
            if(full_line[i] == ",")
            {
                visitors[i]++;
            }
        }
        if (visitors[i] < traffic)
        {
            traffic = visitors[i];
            j = i;
        }
        i++;
    }
    cout << days[j] << " is the busiest day of the week at the motel." << endl;

    return 0;
}
```

#include <fstream>

vis[5]

repeat

should be char

Name:

Date:

Spring 2024

CSCI 1300: Recitation 8

Problem 1.3. The program appends and prepends underscores for every word in the given message string. Assume the message is maximum 4 words. Identify the error(s) in the code below, and write the correct line(s).

```
#include <iostream>
#include <string>
#include <cassert>
using namespace std;

string appendPrepend(string message)
{
    const int LENGTH = 4;
    string message_fragments[LENGTH] = {};
    string empty_word = "";
    split(message, ' ', message_fragments[], LENGTH);
    assert(message_fragments[4] == empty_word);

    string answer, word;
    for(int i = 0; i < LENGTH; i++)
    {
        answer += "_" + message + "_";
    }
    int first_word_length = message_fragments[0].length();
    int second_word_length = message_fragments[1].length();
    assert(message_fragments[1] == answer.substr(first_word_length+3, second_word_length));

    return answer;
}

int main()
{
    cout << "Please enter the string message:" << endl;
    string message;
    getline(cin, message);
    cout << appendPrepend(message);
}
```

No brackets

Add split func

Name:

Date:

Spring 2024

CSCI 1300: Recitation 8

2 Valid Double

Design a function `validateDouble` that accepts a string input and determines if it represents a valid double by iterating through the string. Valid doubles can start with a negative and can have up to one decimal point. Your program should ask the user to input a double, store it as a string and then invoke the `validateDouble` function to check its validity. The program should then print whether the string is a valid double or not. (Negative double are also valid doubles. You can reuse some parts of your `validateInt` function from recitation 5).

Function: validateDouble(string)	`bool validateDouble(string input)`
Purpose:	Iterate through a string and verify if it is a valid double or not.
Parameters:	**input** - The string to be verified
Return value:	It returns true is the string is a valid double. Otherwise returns false.
Error handling/ Boundary conditions:	If length of input = 0, false is returned
Example:	

Sample Code 1:

```
// Assume the proper libraries are included.
// Assume the proper implementation of
↪ validateDouble() is included.

int main()
{
    string number;
    cout << "Enter the double : " << endl;
    getline(cin, number);
    if(!validateDouble(number))
    {
        cout << "The entered string is not a valid
        ↪ double!!" << endl;
    }
    else
    {
        cout << "The entered string is a valid
        ↪ double!!" << endl;
    }
    return 0;
}
```

Sample Output 2.1:

```
Enter the double :
-123.4
The entered string is a valid double!!
```

Here are a few additional sample runs:

Name:

Date:

CSCI 1300: Recitation 8

Spring 2024

Sample Output 2.2:

```
Enter the double :
-12
The entered string is a valid double!!
```

Sample Output 2.3:

```
Enter the double :
23.56e
The entered string is not a valid double!!
```

Sample Output 2.4:

```
Enter the double :
32 56
The entered string is not a valid double!!
```

Sample Output 2.5:

```
Enter the double :
-.
The entered string is not a valid double!!
```

Problem 2.1. Write out the steps you would use to solve this problem by hand as pseudocode. Stating your modifications to your validateInt is also sufficient.

Check if input[0] is '-'
if so, erase it using erase method.
set bool Point found to 0
iterate through input
if c = whitespace or non (nuber or Point)
or c = '.' when Point is found.
return false
else return true.

Name:

Date:

Spring 2024

CSCI 1300: Recitation 8

Problem 2.2. Pick three possible inputs for your program. Try to pick values that will test different aspects of your function. Follow the steps you wrote for these values to find your result, and verify it.

0 → true

-5.7 → true

-5.7.5 → false

Problem 2.3. Translate each of the sample inputs and expected outputs you created into assert statements.

```
assert(isDouble(0) == true)
assert(isDouble(-5.7) == true)
assert(isDouble(-5.7.5) == false)
```

Problem 2.4. Implement your solution in C++ using VS Code. Revise your solution, save, compile and run it again. Are you getting the expected result and output? Keep revising until you do. Make sure you test for the values used in your sample runs, and for the boundary conditions.

Name:

Date:

Spring 2024

CSCI 1300: Recitation 8

3 Center of Mass

The file `coordinates1.txt` contains a list of comma-separated X, Y and Z coordinates for a given geometric body in each column respectively. Your goal is to find the center of mass of the body by computing the average of the X, the Y and the Z coordinates. In order to do this, you must:

1. Read each line in as a string,
2. Use your `split()` function from Homework 5 to separate each line at the commas,
3. Use your `validateDouble()` function from Q2 to validate your strings translate your strings into doubles (you can use `stod` to translate valid strings to doubles).

Report the average of your X coordinates, Y coordinates and Z coordinates as your center of mass. Example output (bold is user input):

Sample Output 3.1:

```
// If file contains two lines:
// 3.25,4.19,-3.56
// 1.04,2.31,5.12
The center of mass is at:  2.145, 3.25, 0.78!
// 2.145 = (3.25+1.04)/2.0
// 3.25 = (4.19+2.31)/2.0
// 0.78 = (-3.56+5.12)/2.0
```

Sample Output 3.2:

```
// If file contains these lines:
// 5.00,0,-0.8
// -3,3.3,-0.75
// 1,-1.0,3.8
// 3.50,0.67,-2
The center of mass is at:  1, 1.65, -0.025!
```

Sample Output 3.3:

```
// If file contains these lines:
// 0,5.00,-0.8
// -3,3.3,0.75
// abc,-1.0,2.8
Invalid value detected!
```

There are several coordinate files included in the week 9 github folder which you can use to test your code. You can also make your own input files.

Name:

Spring 2024

Date:

CSCI 1300: Recitation 8

Problem 3.1. Write out the steps you would use to solve this problem by hand as pseudocode.

Open input file
check if successful
declare vars
double X_mass, Y_mass, Z_mass
make array of strings
declare string var
loop through inputfile using getline
count every iteration in (int num_points)
split string with ',' as delimitor and
array as array and 3 as size.
iterate through array to see if all
values are valid doubles using isdouble.
if no, return -1.

add X-coord to X_mass
add Y-coord to Y_mass
add Z-coord to Z-mass
for every iteration.
calculate $\bar{X}, \bar{Y}, \bar{Z}$
Cout values.

Problem 3.2. Write three possible lines you can include in your file to later test your program. Try to pick values that will test different aspects of your program. Follow the steps you wrote for these values to find your result, and verify it.

{-1, -2, -3 → -1, -2, -3
{a, b, c → invalid value
{700, 700, 700
 0, 0, 0 → 350, 350, 350

Problem 3.3. Implement your solution in C++ using VS Code. Revise your solution, save, compile and run it again. Are you getting the expected result and output? Keep revising until you do. Make sure you test for the values used in your sample runs, and for the boundary conditions. Use the coordinates files on github to test your code.

Problem 3.4. OPTIONAL ADDITIONAL CHALLENGE: create a struct to store a 3D point (i.e., a structure with a double X, Y and Z coordinate) and modify your code to use this structure for the 3D points you read in.

Submission Instructions: Create a zip file that contains your solution .cpp file for question 2, question 3, and photos of this handout and submit on Canvas.