Object Oriented Programming

Fall 2022

Lab 06	
Topic	**Operators Overloading + Friend Functions**

Task 1 :

Run the given below code and see the output. You have to convert 24 hour time format into the 12 hours format for example is the time is 19:39:02 then you have to convert it in 12 hours time format like 07:39:02PM

//////////////////////// main file////////////////////////////////

```cpp
#include<iostream>
#include "TimeTeller.h"

using namespace std;

int main() {

    TimeTeller obg1;

    //TimeTeller obj2(5);
    TimeTeller obj2(5, 15, 30);
    system("pause");
}
```

//////////// Function definitions files////////////////////////////

```cpp
#include <ctime>
#include <iostream>
#include "TimeTeller.h"

using namespace std;

TimeTeller::TimeTeller() {

    cout << "Constructor Without Parameters: " << endl;
    hh = mm = ss = 0;

    // current date/time based on current system
```

```cpp
        time_t now = time(0);

        // convert now to string form
        const int bufferSize = 30;
        char time_buff[bufferSize];

        //ctime_s(char  * buff, rsize_t buffSize, const time_t * sourceTime );
        ctime_s(time_buff, bufferSize, &now);

        long long total = now;

        long hrs = (long) total / 3600;
        hrs = (hrs % 24) + 5;
        total %= 3600;
        int mnt = (long)total / 60;
        total %= 60;
        int snd = (long) total;

        hh = hrs;
        mm = mnt;
        ss = snd;

        cout << "The local seconds: " << now << endl;
        cout << "The local date and time is: " << time_buff << endl;
        cout << "Hours: " << hrs << " Minutes: " << mnt << " Seconds: " << snd << endl;
}

TimeTeller::TimeTeller(int h=0, int m=0, int s=0) {

        cout << "Parameterized Constructor:" << endl;
        hh = h;
        mm = m;
        ss = s;
}



///////////////Header file//////////

#pragma once
class TimeTeller
{
        int hh;
        int mm;
        int ss;

public:
        TimeTeller();
        TimeTeller(int, int, int);
};
```

```
■ D:\time\Debug\time.exe

Constructor Without Parameters:
The local seconds: 1669300742
The local date and time is: Thu Nov 24 19:39:02 2022

Hours: 19 Minutes: 39 Seconds: 2
Parameterized Constructor:
Press any key to continue . . . ▁
```

Task 2 :

Create a class called **Fraction** for performing arithmetic with fractions. Write a driver program to test your class. Use integer variables to represent the private data of the class, the numerator and the denominator. Provide a constructor function that enables an object of this class to be initialized when it is declared.

The constructor should contain default values in case no initializers are provided and should store the fraction in reduced form (use a utility function **reduction**() function to reduce the result for example the fraction 2/4 would be stored in the object as 1 in the numerator and 2 in the denominator). Your results should be in reduced forms.

You have to implement functionalities to:

1.   Add fractions using the operator +
2.   Subtract fractions using the operator –
3.   Multiply fractions using the operator *
4.   Divide fractions using the operator /
5.   Compare fractions (five functions) using operators **==, >, <, >=, <=**
6.   Increment fractions using post-increment and pre-increment operators ++
7.   Decrement fractions using post-decrement and pre-decrement operators - -
8.   Take negative of fractions using operator unary –. Also overload the fraction unary +
9.   Compound operators += , -=, *= ,/=
10. Display fractions using the operator <<
11. Input fractions using the operator >>
12. Copy contents of a fraction in another fraction using operator =

Task 3: Extra Bonus Task

A main() function is given below. Based on this function, your task is to write complete code required to run the main() function successfully and produce the exact output given. **You are not allowed to return any memory handler anywhere in your entire code.**
Restrictions: Just run main() function given below as it is. Do not change even a single character of this main.

```cpp
int main()
{
    MyString str1;// make a default string of 100 size
    str1.add('[');//insert at position 1 or index 0
    str1.add('A');//insert at position 2 or index 1
    str1.add('B');//insert at position 3 or index 2
    str1.add('C');//insert at position 4 or index 3
    str1.add('D');//insert at position 5 or index 4
    str1.add('E');//insert at position 6 or index 5
    str1.add('F');//insert at position 7 or index 6
    str1.add('G');//insert at position 8 or index 7
    str1.add('h');//insert at position 9 or index 8
    str1.add('i');//insert at position 10 or index 9
    str1.add('j');//insert at position 11 or index 10
    str1.add('k');//insert at position 12 or index 11
    str1.add('l');//insert at position 13 or index 12
    str1.add('m');//insert at position 14 or index 13
    str1.add('n');//insert at position 15 or index 14
    str1.add('o');//insert at position 16 or index 15
    str1.add('p');//insert at position 17 or index 16
    str1.add(']');//insert at position 18 or index 17

    cout << "str1\n";
    print(str1);
    cout << "Lenghth of str1 is = " << str1.length() << endl;

    cout << "str2\n";
    MyString str2(str1, 2, 9);//should copy str1 to str2 from position 2 to 9
    print(str2);
    cout << "Lenghth of str2 is = " << str2.length() << endl;

    cout << "str4\n";
```

```cpp
        const MyString str4; //It should accesss all constant functions
        print(str4);
        cout << "Lenghth of str4 is =" << str4.length() << endl;

        str1.clear();//It should clear the string by placing null char at all indexes and
        setting count to zero
        cout << "After clearing str1\n";
        print(str1);

        str1.add('$');
        cout << "After adding $ to str1\n";
        print(str1);

        MyString str3;
        str3 = str1 + str2;//It will concatenate both the strings and assign it to str3
        cout << "str3\n";
        print(str3);

        MyString str5(5);//should make an empty array of size
5       str5.add('a');
        str5.add('B');
        str5.add('c');
        str5.add('D');
        str5.add('e');
        cout << "str5\n";
        print(str5);

        str5[0] = str5[1] + 1; //add 1 to the char at 1st index and assign it to 0th index
        print(str5);
        print(str5++);//post-increment
        print(++str5);//pre-increment
        print(--str5);//pre-decrement
        print(str5--);//post-decrement

        MyString str8;
        str8 = str5;//assigning str5 to str8
        cout << "str8\n";
        print(str8);
        return 0;
}
```

Output:

```
Microsoft Visual Studio Debug Console
str1
[ A B C D E F G h i j k l m n o p ]
Lenghth of str1 is = 18
str2
A B C D E F G h
Lenghth of str2 is = 8
str4
String is empty.
Lenghth of str4 is =0
After clearing str1
String is empty.
After adding $ to str1
$
str3
$ A B C D E F G h
str5
a B c D e
C B c D e
C B c D e
E D e F g
D C d E f
D C d E f
str8
C B c D e

C:\Users\mbilal.ishfaq\source\repos\
Press any key to close this window .
```

Now add the following lines of code in the main() function too and see if it gives
the output given below:

```
//Now test these lines in your main too, and check if it provides you the following
output or not
        MyString str9;
        str9.add('B');
        str9.add('I');
        str9.add('L');
        str9.add('A');
```

```
str9.add('L');
str9.add(' ');
str9.add('I');
str9.add('S');
str9.add('H');
str9.add('F');
str9.add('A');
str9.add('Q');

cout << "str9\n";
++++str9;  //check with return by value and return by reference both
print(str9);

MyString str10;
MyString str11;

(str11=str10)=str9;  //check with return by value and return by reference both

cout <<"str10\n";
print(str10);

cout << "str11\n";
print(str11);
```

Added Output:

If it is not giving you the above output then there is something wrong with your code. Correct it! And check in above code that return by reference and return by value gives same output against ++operator and = operator or not.

Now answer the following question with an example code?

1) Why we should always return copy-assignment operator(=),preincrement(++) and pre-decrement(--) operators by reference ?

BONUS TASK:

Make your class able to run the following lines of code.

```
MyString str12{ 'a', 'b', 'c','d'};
print(str12);
str12 = { 'e', 'f', 'g', 'h' , 'i' };
print(str12);
```