Assignment 27

1. Define a class Complex with appropriate instance variables and member functions.

Define following operators in the class:

a. +

b. -

c. \*

d. ==

#include<iostream>

using namespace std;

class complex

{   private:

        int real;

        int img;

    public:

        void set(int x, int y)

        {

            real=x;

            img=y;

        }

        complex()

        {

           // cout<<"default constructor called"<<endl;

        }

        void show()

        {

            cout<<real<<" + "<<img<<" i"<<endl;

        }

        complex operator+(complex c)

        {

            complex temp;

            temp.real=real+c.real;

            temp.img= img+c.img;

            return temp;

        }

        friend complex operator-(complex c1, complex c2);

        friend int operator==(complex c1, complex c2);

       complex operator\*(complex c)

       {

            complex temp;

            temp.real= real\* c.real;

            temp.img= img\*c.img;

            return temp;

       }

};

complex operator-(complex c1, complex c2)

{

    complex c;

    c.real= c1.real - c2.real;

    c.img = c1.img - c2.img;

    return (c);

}

int operator==(complex c1, complex c2)

{

    if(c1.real==c2.real  && c1.img==c2.img)

        return 1;

    else

        return 0;

}

int main()

{

    complex c1, c2,sum, sub, mul;

    c1.set(2,3);

    c2.set(1,-2);

    sum=c1+c2;

    c1.show();

    c2.show();

    cout<<"----------------"<<endl;

    sum.show();

    cout<<"subtraction of the complex number is : "<<endl;

    sub= c1-c2;

    sub.show();

    cout<<"----------------"<<endl;

    cout<<"Mulitplication of the two complex nuber is :"<<endl;

    mul=c1\*c2;

    mul.show();

    cout<<"-------------------"<<endl;

    cout<<"check both the complex numbers are equal : "<<endl;

    if(c1==c2)

        cout<<" Result = yes"<<endl;

    else

        cout<<"Result =  no "<<endl;

    return 0;

}

output:

c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ operator\_overloading.cpp -o operator\_overloading } ; if ($?) { .\operator\_overloading }

2 + 3 i

1 + -2 i

----------------

3 + 1 i

subtraction of the complex number is :

1 + 5 i

----------------

Mulitplication of the two complex nuber is :

2 + -6 i

-------------------

check both the complex numbers are equal :

Result = no

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

2. Write a C++ program to overload unary operators that is increment and decrement.

3. Write a C++ program to add two complex numbers using operator overloaded by a

friend function.

#include<iostream>

using namespace std;

class complex

{

    private:

        int real;

        int img;

    public:

        complex()

        {

            real=0;

            img=0;

        }

        void set(int x,int y)

        {

            real= x;

            img=y;

        }

        void showData()

        {

            cout<<real<<" + "<<img<<"i"<<endl;

        }

        friend complex operator+(complex c1, complex c2);

};

complex operator+(complex c1, complex c2)

{

    complex c;

    c.real= c1.real + c2.real;

    c.img= c1.img + c2.img;

    return c;

}

int main()

{

    complex c1,c2,c3;

    c1.set(2,5);

    c2.set(6,4);

    c1.showData();

    c2.showData();

    c3=c1+c2;

    cout<<"addition of two complex number is : "<<endl;

    c3.showData();

    return 0;

}

Ouput:

"c:\Users\tusha\Documents\coadind\" ; if ($?) { g++ friend\_addition.cpp -o friend\_addition } ; if ($?) { .\friend\_addition }

2 + 5i

6 + 4i

addition of two complex number is :

8 + 9i

PS C:\Users\tusha\Documents\coadind>

4. Create a class Time which contains:

- Hours

- Minutes

- Seconds

Write a C++ program using operator overloading for the following:

1. = = : To check whether two Times are the same or not.

2. >> : To accept the time.

3. << : To display the time.

#include<iostream>

using namespace std;

class time

{

    private:

        int hr;

        int min;

        int sec;

    public:

        time()

         {

            hr=0;

            min=0;

            sec=0;

         }

        friend int operator>>(istream &input, time &t)

        {

            cout<<"\n enter the hours: ";

            input>>t.hr;

            cout<<"enter minutes: ";

            input>>t.min;

            cout<<"enter sec : ";

            input>>t.sec;

            t.min=t.min+t.sec/60;

            t.sec= t.sec%60;

            t.hr=t.hr+t.min/60;

            t.min=t.min%60;

            if(t.hr>=25)

                return 1;

            else

                return 0;

        }

        friend void  operator<<(ostream &output, time t)

        {

            output<<"\n Hrs : "<<t.hr;

            output<<"\n Min : "<<t.min;

            output<<"\n sec : "<<t.sec;

        }

        int operator==(time t)

        {

            int tol1= hr \* 3600 + min \* 60 + sec;

            int tol2= t.hr \* 3600 + t.min \* 60 + t.sec;

            if(tol1 == tol2)

                return 1;

            else

                return 0;

        }

        ~time(){}

};

int main()

{

    time t1,t2;

    cout<<"enter the 1st time: "<<endl;

    cout<<"-----------------------------"<<endl;

    if(cin>>t1)

    {

        cout<<"invalid time"<<endl;

        return 0;

    }

    cout<<"\n first time : "<<endl;

    cout<<t1;

    cout<<"\n enter the second time : "<<endl;

    cout<<"----------------------------------"<<endl;

    if(cin>>t2)

    {

        cout<<"invalid time"<<endl;

        return 0;

    }

    cout<<"\n second time : "<<endl;

    cout<<t2;

    cout<<"\n---------------------"<<endl;

    if(t1==t2)

        cout<<"\n both the tiems are same"<<endl;

    else

        cout<<"\n both the times are different"<<endl;

    return 0;

Output;

cd "c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ time\_overloading.cpp -o time\_overloading } ; if ($?) { .\time\_overloading }

enter the 1st time:

-----------------------------

enter the hours: 12

enter minutes: 23

enter sec : 45

first time :

Hrs : 12

Min : 23

sec : 45

enter the second time :

----------------------------------

enter the hours: 12

enter minutes: 23

enter sec : 45

second time :

Hrs : 12

Min : 23

sec : 45

---------------------

both the tiems are same

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

5. Consider following class Numbers

class Numbers

{

int x,y,z;

public:

// methods

};

#include<iostream>

using namespace std;

class Numbers

{

    int x,y,z;

    public:

        Numbers()

        {

            x=y=z=0;

        }

        void accept()

        {

            cout<<"enter the three numbers : ";

            cin>>x>>y>>z;

        }

        void display()

        {

            cout<<x<<", "<<y<<", "<<z<<endl;

        }

        void operator-()

        {

            x=-x;

            y=-y;

            z=-z;

        }

};

int main()

{

    Numbers num;

    num.accept();

    cout<<"the numbers are : "<<endl;

    num.display();

    -num;

    cout<<"negative numbers are : "<<endl;

    num.display();

    return 0;

}

Overload the operator unary minus (-) to negate the numbers.

c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ unary\_minus.cpp -o unary\_minus } ; if ($?) { .\unary\_minus }

enter the three numbers : 12 45 78

the numbers are :

12, 45, 78

negative numbers are :

-12, -45, -78

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

6. Create a class CString to represent a string.

a) Overload the + operator to concatenate two strings.

b) == to compare 2 strings.

#include<iostream>

#include<string.h>

using namespace std;

class Cstring

{

    private:

        char str[20];

    public:

        void get\_string()

        {

            cout<<"\n enter the string    :   ";

            gets(str);

        }

        void display()

        {

            cout<<str<<endl;

        }

        Cstring operator+(Cstring s)

        {

            Cstring temp;

            strcpy(temp.str, str);

            strcat(temp.str, s.str);

            return temp;

        }

        int operator==(Cstring &s)

        {

            {

               int x= strcmp(str , s.str);

               if(x==0)

                    return 1;

               else

                    return 0;

            }

        }

};

int main()

{

    Cstring s1, s2,s3;

    s1.get\_string();

    s2.get\_string();

    s3=s1+s2;

    cout<<"------------------------"<<endl;

    s3.display();

    cout<<"-----------------------"<<endl;

    if(s1==s2)

        cout<<"both the string are same "<<endl;

    else

        cout<<"strings are different "<<endl;

    cout<<"---------------------------"<<endl;

    return 0;

}

cd "c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ string\_concatenating.cpp -o string\_concatenating } ; if ($?) { .\string\_concatenating }

enter the string : tushar

enter the string : tushar

------------------------

tushartushar

-----------------------

both the string are same

---------------------------

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

7. Define a C++ class fraction

class fraction

{

long numerator;

long denominator;

Public:

fraction (long n=0, long d=0);

}

Overload the following operators as member or friend:

a) Unary ++ (pre and post both)

b) Overload as friend functions: operators << and >>.

#include<iostream>

using namespace std;

class fraction

{

    private:

        int num;

        int deno;

    public:

        fraction()

        {

            num=0;

            deno=0;

        }

        fraction(int n, int d)

        {

            num=n;

            deno=d;

        }

        friend void operator>>(istream &input, fraction &f)

        {

            cout<<"\n enter the numerator :  ";

            input>>f.num;

            cout<<"\n enter the denominator :  ";

            input>>f.deno;

        }

        friend void operator<<(ostream &output, fraction &f)

        {

            output<<f.num<<"/"<<f.deno<<endl;

        }

        fraction operator++()

        {

            ++num;

            ++deno;

            return (\*this);

        }

        fraction operator++( int s)

        {

            fraction temp;

            temp.num++;

            temp.deno++;

            return temp;

        }

};

int main()

{

    fraction f1, f2;

    cout<<"\n f1  :  ";

    cout<<f1;

    cout<<"\n f2  :  ";

    cout<<f2;

    cout<<"---------------------";

    cout<<"\n enter the first fraction value :  ";

    cin>>f1;

    cout<<"\n ++f1  :  ";

    ++f1;

    cout<<f1;

    cout<<"\n f1++  :  ";

    f1++;

    cout<<f1;

    cout<<"----------------------";

    cout<<"\n enter the second fraction value  :  ";

    cin>>f2;

    f2= ++f1;

    cout<<"\n f2= ++f1 ";

    cout<<"\n f1  :  ";

    cout<<f1;

    cout<<"\n f2  :  ";

    cout<<f2;

    f2=f1++;

    cout<<"\n f2=f1++ ";

    cout<<"\n f1  :  ";

    cout<<f1;

    cout<<"\n f2  :  ";

    cout<<f2;

    return 0;

}

Output-

PS C:\Users\tusha\Documents\coadind> cd "c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ fraction\_incement.cpp -o fraction\_incement } ; if ($?) { .\fraction\_incement }

f1 : 0/0

f2 : 0/0

---------------------

enter the first fraction value :

enter the numerator : 2

enter the denominator : 5

++f1 : 3/6

f1++ : 3/6

----------------------

enter the second fraction value :

enter the numerator : 3

enter the denominator : 4

f2= ++f1

f1 : 4/7

f2 : 4/7

f2=f1++

f1 : 4/7

f2 : 1/1

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

8. Consider a class Matrix

Class Matrix

{

int a[3][3];

Public:

//methods;

};

Overload the - (Unary) should negate the numbers stored in the object.

#include<iostream>

using namespace std;

class matrix

{

    int a[3][3];

    public:

        void get\_values()

        {

            cout<<"enter the valus of matrix (3 x 3)"<<endl;

            for(int i=0; i<3; i++)

            {

                for(int j=0; j<3; j++)

                {

                    cin>>a[i][j];

                }

            }

        }

        void display()

        {

            for(int i=0; i<3; i++)

            {

                for(int j=0; j<3; j++)

                {

                    cout<<a[i][j]<<"  ";

                }

                cout<<endl;

            }

        }

        void operator-()

        {

            for(int i=0; i<3; i++)

            {

                for(int j=0; j<3; j++)

                {

                    a[i][j] = -a[i][j];

                }

            }

        }

};

int main()

{

    matrix m;

    m.get\_values();

    cout<<"--------------------"<<endl;

    cout<<"\n matrix is : "<<endl;

    m.display();

    cout<<"--------------------"<<endl;

    -m;

    cout<<"\n matrix is : "<<endl;

    m.display();

    return 0;

}

Output –

c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ matrix.cpp -o matrix } ; if ($?) { .\matrix }

enter the valus of matrix (3 x 3)

1 2 3 4 5 6 7 8 9

--------------------

matrix is :

1 2 3

4 5 6

7 8 9

--------------------

matrix is :

-1 -2 -3

-4 -5 -6

-7 -8 -9

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

9. Consider the following class mystring

Class mystring

{

char str [100];

Public:

// methods

};

#include<iostream>

#include<string.h>

using namespace std;

class mystring

{

    char str[100];

    public:

        void get\_string()

        {

            cout<<"enter the string :  ";

            gets(str);

        }

        void display()

        {

            cout<<str;

        }

        void operator!();

};

void mystring::operator!()

{

    for (int i = 0; i < strlen(str); i++)

    {

        if(str[i]>='a' && str[i]<='z')

        {

            str[i]=str[i]-32;

        }

        else if(str[i]>='A' && str[i]<='Z')

            str[i]=str[i]+32;

    }

    cout<<str<<endl;

}

int main()

{

    mystring s;

    s.get\_string();

    cout<<"entered string before modification : "<<endl;

    s.display();

    cout<<"\n-----------"<<endl;

    cout<<"entered string after modification : "<<endl;

    !s;

    cout<<"--------------------"<<endl;

    return 0;

}

Overload operator “!” to reverse the case of each alphabet in the string

(Uppercase to Lowercase and vice versa).

Output:

$?) { .\lower\_to\_upper }

enter the string : tushar

entered string before modification :

tushar

-----------

entered string after modification :

TUSHAR

--------------------

PS C:\Users\tusha\Documents\coadind\assignemet\_27>

10.Class Matrix

{

int a[3][3];

Public:

//methods;

};

Let m1 and m2 are two matrices. Find out m3=m1+m2 (use operator

overloading).

#include<iostream>

using namespace std;

class matrix

{

    int a[3][3];

    public:

        void get\_values()

        {

            cout<<"enter the values of matrix "<<endl;

            for (int i = 0; i < 3; i++)

            {

                for (int j = 0; j < 3; j++)

                {

                    cin>>a[i][j];

                }

            }

        }

        void display()

        {

            for (int i = 0; i < 3; i++)

            {

                for (int j = 0; j < 3; j++)

                {

                    cout<<a[i][j]<<"  ";

                }

                cout<<endl;

            }

        }

        matrix operator+(matrix m)

        {

            matrix temp;

            for (int i = 0; i < 3; i++)

            {

                for (int j = 0; j < 3; j++)

                 {

                    temp.a[i][j]=a[i][j]+m.a[i][j];

                 }

            }

            return temp;

        }

};

int main()

{

    matrix m1, m2, m3;

    m1.get\_values();

    m2.get\_values();

    cout<<"------------"<<endl;

    cout<<"entered matrix 1 is : "<<endl;

    m1.display();

    cout<<"-----------"<<endl;

    cout<<"entered matrix 2 is : "<<endl;

    m2.display();

    cout<<"-----------------------"<<endl;

    cout<<"addition of the two matrices is : "<<endl;

    m3=m1+m2;

    m3.display();

    return 0;

}

Output:

c:\Users\tusha\Documents\coadind\assignemet\_27\" ; if ($?) { g++ matrix\_sum.cpp -o matrix\_sum } ; if ($?) { .\matrix\_sum }

enter the values of matrix

1 2 3 4 5 6 7 8 9

enter the values of matrix

9 8 7 6 5 4 3 2 1

------------

entered matrix 1 is :

1 2 3

4 5 6

7 8 9

-----------

entered matrix 2 is :

9 8 7

6 5 4

3 2 1

-----------------------

addition of the two matrices is :

10 10 10

10 10 10

10 10 10

PS C:\Users\tusha\Documents\coadind\assignemet\_27>