**Lab 4.Operator Overloading**

**Exercises**

**1.Create a class called string. Overload the operators + and += so that statements like**

**s3 = s1 + s2 ;**

**s3 +=s2**

**can be used.**

**Ans.**

/\*program to overload + & += opertor in string\*/

#include<iostream>

using namespace std;

//Declaring class

class String{

private:

string name;

public:

String(){}

String(string n)

{

name = n;

}

//member function

void display(){

cout<<" Name : "<<name<<endl;

}

//Overloaded +operator in string

String operator +(String &s)

{

String x;

x.name = name + s.name; //concanating string

return x;

}

//Overloaded +=operator

String operator +=(String &s)

{

String x;

x.name = name += s.name;

return x;

}

};//end of class

//main program

int main()

{

String s1("Raban"),s2(" Kathariya"),s3;

//overloading + operator

s3 = s1+s2;

cout<<"After overloading + operator \n";

//displaying after overloading

s3.display();

cout<<"\n\n";

//overloading += operator

s3 +=s2;

cout<<"After overloading += operator \n";

//displaying after overloading

s3.display();

}

**2.Create a class called Person that has a member variable called age, in the public section it has necessary constructor to assign value to the member variable and member function to display the age of person**

**Now add another member function by overloading the <= operator for comparing the ages of two Person objects**

**In main create two objects of the type Person, use the overloaded <= operator function** **to compare the ages of two persons**

**Ans**

/\* program to overloaded <= operator function to compare the ages of two persons \*/

#include<iostream>

using namespace std;

//Declaring class

class person

{

private:

int age;

public:

// Default constructor

person(){}

// Parameterised constructor

person(int a )

{

age = a;

}

//methods to display age of person

void display()

{

cout<<"Age of P1 is : "<<age<<endl;

}

void display1()

{

cout<<"Age of P2 is : "<<age<<endl;

}

//overloaded <= opertor

bool operator <=(person &p)

{

if(age<=p.age)

{

return true;

}

else

{

return false;

}

}

}; //end of class

//Main program

int main(){

//Creating two object

person p1(100),p2(10);

//overloading <= operator

if(p1 <= p2)

{

p1.display();

p2.display1();

cout<<"P1 is less than or equal to P2"<<endl;

}

else

{ p1.display();

p2.display1();

cout<<"P2 is less than or equal to P1"<<endl;

}

}

**3.Create a class called Integer that has a data member called num and perform the following tasks:**

* **Assume necessary constructors to initialize the data member and necessary member functions to display the value stored in the data member num.**
* **Overload the operators +, - and \* using operator function to add , subtract and multiply two numbers.**
* **In main create five objects of class Integer called num1, num2, num3, num4 and num5 pass necessary values so that the data member of the objects get initialized.**
* **Use the overload operators to evaluate the following expression:**

**num5 = num2 \* num1 + num3 - num4**

**Ans.**

/\* Program to evaluate the expression using +,-, and \* operator\*/

#include<iostream>

using namespace std;

//Declaring class

class integer

{

private:

int num;

public:

integer(){} // Default constructor

integer(int n ) // Parameterised constructor

{

num = n;

}

//method to display age of person

void displayAdd()

{

cout<<"After overloading + operator ,result = "<<num<<endl;

}

void displaySub() //method to display age of person

{

cout<<"After overloading - operator ,result = "<<num<<endl;

}

void displayMul() //method to display age of person

{

cout<<"After overloading \* operator ,result = "<<num<<endl;

}

void display()

{

cout<<"After overloading +,-& \* operator ,result = "<<num;

}

//oveloaded +operator

integer operator +(integer d)

{

integer x;

x.num = num + d.num;

return x;

}

//overloaded - opertor

integer operator- (integer d)

{

integer x;

x.num = num - d.num;

return x;

}

//overloaded \* operator

integer operator \*(integer d)

{

integer x;

x.num = num \* d.num;

return x;

}

}; //end of class

int main()

{

//creating object

integer num1(10),num2(2),num3(2),num4(1),num5;

//Overloading all operators in single line

num5 = num2\*num1+num3-num4;

num5.display();

return 0;

}

**4.Design a class called NumDays. The classs purpose is to store a value that represents a number of work hours and convert it to a number of days. For example, 8 hours would be converted to 1 day, 12 hours would be converted to 1.5 days, and 18 hours would be converted to 2.25 days. The class should have a constructor that accepts a number of hours, as well as member functions for storing and retrieving the hours and days. The class should also have the following overloaded operators:**

**+ Addition operator. When two NumDays objects are added together, the overloaded + operator should return the sum of the two objects hours members. - Subtraction operator. When one NumDays object is subtracted from another, the overloaded - operator should return the difference of the two objects hours members.++ Prefix and postfix increment operators. These operators should increment the number of hours stored in the object. When incremented, the number of days should be automatically recalculated.-- Prefix and postfix decrement operators. These operators should decrement the number of hours stored in the object. When decremented, the number of days should be automatically recalculated.**

**Ans.**

/\*program to overload +,-,++,-- operator \*/

#include<iostream>

using namespace std;

//Declaring class

class NumDays

{

private:

int hours;

public:

// Default constructor

NumDays (){}

// Parameterised constructor

NumDays (int h)

{

hours = h;

}

void display()

{

cout<<"No. Of Hours = "<<hours<<endl;

cout<<hours<<" is equal to "<<0.125\*hours<<" days"<<endl;

}

//overloaded + operator

NumDays operator +(NumDays &d)

{

NumDays x;

x.hours = hours + d.hours;

return x;

}

//overloaded - operator

NumDays operator -(NumDays &d)

{

NumDays x;

x.hours = hours - d.hours;

return x;

}

//overloaded ++ prefix operator

NumDays operator++()

{

NumDays d;

d.hours = ++hours;

return d;

}

//overloaded postfix ++ operator

NumDays operator++(int)

{

NumDays n;

n.hours = hours++;

return n;

}

};//end of class

//main program

int main()

{

NumDays t1(8) , t2(4) , t3;

//overloading + operator

cout<<"After overlading + operator\n";

t3 = t1+t2;

t3.display();

cout<<"\n\n";

cout<<"After overlading - operator\n";

t3 = t1-t2;

t3.display();

cout<<"\n\n";

//overloading ++ operator

cout <<"After prefix ++ operator increment of t1\n";

t3 = ++t1;

t3.display();

cout<<"\n\n";

//overloading ++ operator

cout <<"After postfix ++ operator increment of t2\n" <<endl;

t3 = t2++;

t3.display();

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***