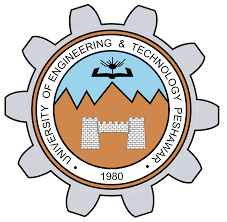
****

**OOP LAB**

**Lab Report No. 03**

**Submitted By: Jamshid Bacha**

**Regesteration No. 16PWCSE1404**

**Submitted To: Engr. Sumayya Salahuddin**

**Section: B**

**Batch: 18**

**Department: CSE**

**Date: 13-10-2017**

**University of Engineering and Technology Peshawar**

**Home Task**

**Task 01:**

#include<iostream>

using namespace std;

class RationalNumber

{

private:

int numerator,denominator;

public:

RationalNumber();

RationalNumber(int nom,int denom);

void AddFraction(RationalNumber num1,RationalNumber num2);

void SubtractFraction(RationalNumber num1,RationalNumber num2);

void MultiplyFraction(RationalNumber num1,RationalNumber num2);

void Division(RationalNumber num1,RationalNumber num2);

void display();

bool isGreater(RationalNumber num1,RationalNumber num2);

bool isSmaller(RationalNumber num1,RationalNumber num2);

bool isGreaterEqual(RationalNumber num1,RationalNumber num2);

bool isSmallerEqual(RationalNumber num1,RationalNumber num2);

bool isEqual(RationalNumber num1,RationalNumber num2);

bool isNotEqual(RationalNumber num1,RationalNumber num2);

};

RationalNumber::RationalNumber()

{

numerator=1;

denominator=1;

}

RationalNumber::RationalNumber(int nom,int denom)

{

numerator=nom;

if(denom>0)

denominator=denom;

else

denominator=1;

}

void RationalNumber::AddFraction(RationalNumber num1,RationalNumber num2)

{

numerator=num1.numerator\*num2.denominator + num2.numerator\*num1.denominator;

denominator=num1.denominator\*num2.denominator;

}

void RationalNumber::SubtractFraction(RationalNumber num1,RationalNumber num2)

{

numerator=num1.numerator\*num2.denominator - num1.denominator\*num2.numerator;

denominator=num1.denominator\*num2.denominator;

}

void RationalNumber::MultiplyFraction(RationalNumber num1,RationalNumber num2)

{

numerator=num1.numerator\*num2.numerator;

denominator=num1.denominator\*num2.denominator;

}

void RationalNumber::Division(RationalNumber num1,RationalNumber num2)

{

numerator=num1.numerator\*num2.denominator;

denominator=num1.denominator\*num2.numerator;

}

void RationalNumber::display()

{

cout<<numerator<<" / "<<denominator<<endl;

}

bool RationalNumber::isGreater(RationalNumber num1,RationalNumber num2)

{

if(((float)num1.numerator/(float)num1.denominator)>((float)num2.numerator/(float)num2.denominator))

return true;

else

return false;

}

bool RationalNumber::isSmaller(RationalNumber num1,RationalNumber num2)

{

if(((float)num1.numerator/(float)num1.denominator)<((float)num2.numerator/(float)num2.denominator))

return true;

else

return false;

}

bool RationalNumber::isGreaterEqual(RationalNumber num1,RationalNumber num2)

{

if(((float)num1.numerator/(float)num1.denominator)>=((float)num2.numerator/(float)num2.denominator))

return true;

else

return false;

}

bool RationalNumber::isSmallerEqual(RationalNumber num1,RationalNumber num2)

{

if(((float)num1.numerator/(float)num1.denominator)<=((float)num2.numerator/(float)num2.denominator))

return true;

else

return false;

}

bool RationalNumber::isEqual(RationalNumber num1,RationalNumber num2)

{

if(((float)num1.numerator/(float)num1.denominator)==((float)num2.numerator/(float)num2.denominator))

return true;

else

return false;

}

bool RationalNumber::isNotEqual(RationalNumber num1,RationalNumber num2)

{

if(((float)num1.numerator/(float)num1.denominator)!=((float)num2.numerator/(float)num2.denominator))

return true;

else

return false;

}

int main()

{

RationalNumber num1(1,3),num2(5,4),num3;

cout<<"First Number Is: "<<endl;

num1.display();

cout<<"Second Number Is: "<<endl;

num2.display();

num3.AddFraction(num1,num2);

cout<<"Addition of Num1 and Num2 is: "<<endl;

num3.display();

num3.SubtractFraction(num1,num2);

cout<<"Subtracion of Num1 and Num2 is: "<<endl;

num3.display();

num3.MultiplyFraction(num1,num2);

cout<<"Multiplication of Num1 and Num2 is: "<<endl;

num3.display();

num3.Division(num1,num2);

cout<<"Division of Num1 and Num2 is: "<<endl;

num3.display();

if(num3.isGreater(num1,num2))

cout<<"Num1 Is Greater then Num2: "<<endl;

if(num3.isSmaller(num1,num2))

cout<<"Num1 Is Smaller then Num2: "<<endl;

if(num3.isGreaterEqual(num1,num2))

cout<<"Num1 Is Greater and Equal then Num2: "<<endl;

if(num3.isSmallerEqual(num1,num2))

cout<<"Num1 Is Smaller Equal then Num2: "<<endl;

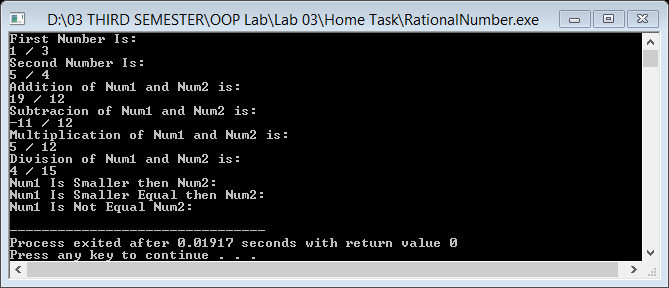
if(num3.isEqual(num1,num2))

cout<<"Num1 is Equal to Num2: "<<endl;

if(num3.isNotEqual(num1,num2))

cout<<"Num1 Is Not Equal Num2: "<<endl;

return 0;

****}

**Task 02:**

#include<iostream>

using namespace std;

class IntegerSet

{

private:

bool Integer[50];

public:

IntegerSet();

IntegerSet unionOfIntegerSets(IntegerSet a,IntegerSet b);

IntegerSet intersectionOfIntegerSets(IntegerSet a,IntegerSet b);

bool isEqualTo(IntegerSet a,IntegerSet b);

void insertElement(int x);

void deleteElement(int y);

void showset();

};

IntegerSet::IntegerSet()

{

for(int i=0;i<50;i++)

{

Integer[i]=0;

}

}

void IntegerSet::insertElement(int x)

{

if(x>=0 && x<50)

{

Integer[x]=1;

}

}

IntegerSet IntegerSet::unionOfIntegerSets(IntegerSet a,IntegerSet b)

{

IntegerSet third\_set;

for(int i=0;i<50;i++)

{

if(a.Integer[i]==1 || b.Integer[i]==1)

{

third\_set.Integer[i]=1;

}

}

return third\_set;

}

IntegerSet IntegerSet::intersectionOfIntegerSets(IntegerSet a,IntegerSet b)

{

IntegerSet third\_set;

for(int i=0;i<50;i++)

{

if(a.Integer[i]==1 && b.Integer[i]==1)

third\_set.Integer[i]=1;

}

return third\_set;

}

void IntegerSet::deleteElement(int y)

{

if(y>=0 && y<50)

{

Integer[y]=0;

}

}

void IntegerSet::showset()

{

for(int i=0;i<50;i++)

{

if(Integer[i]==1)

{

cout<<" "<<i;

}

}

cout<<endl;

}

bool IntegerSet::isEqualTo(IntegerSet a,IntegerSet b)

{

for(int i=0;i<50;i++)

{

if(a.Integer[i]!=b.Integer[i])

return false;

else

return true;

}

}

int main()

{

cout<<"Enter the Set element to finish enter -1: "<<endl;

int element;

IntegerSet a , b , c ,d;

while(cin>>element,element!=-1)

{

a.insertElement(element);

}

cout<<"Set A: ";

a.showset();

cout<<endl<<"Enter Second Set element: "<<endl;

element=0;

while(cin>>element,element!=-1)

{

b.insertElement(element);

}

cout<<"Set B: ";

b.showset();

cout<<endl<<"The Union of set A and set B is U: ";

c=c.unionOfIntegerSets(a,b);

c.showset();

cout<<endl<<"The Intersection of set A and set B is I: ";

d=d.intersectionOfIntegerSets(a,b);

d.showset();

if(c.isEqualTo(a,b))

cout<<endl<<"Set A and Set B are Equal: "<<endl;

else

cout<<endl<<"Set A and Set B are not Equal: "<<endl;

cout<<endl<<"IF you want to delet some element from Set A then enter it: "<<endl;

element=0;

while(cin>>element,element!=-1)

{

a.deleteElement(element);

}

cout<<"After deleting some element from Set A: ";

a.showset();

cout<<endl<<"IF you want to delet some element from Set B then enter it: "<<endl;

element=0;

while(cin>>element,element!=-1)

{

b.deleteElement(element);

}

cout<<"After deleting some element from Set B: ";

b.showset();

cout<<endl<<"The Union of set A and set B is U: ";

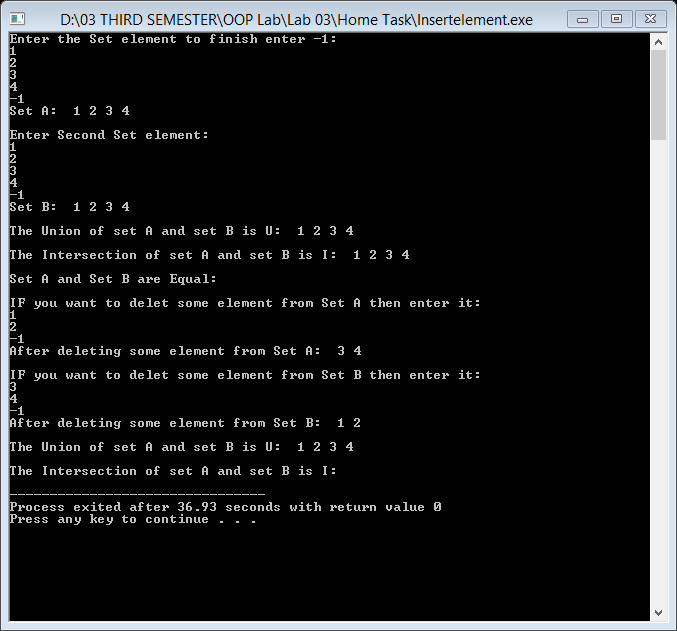
c=c.unionOfIntegerSets(a,b);

c.showset();

cout<<endl<<"The Intersection of set A and set B is I: ";

d=d.intersectionOfIntegerSets(a,b);

d.showset();

****}