**Friend Function and Friend Class**

**LAB # 07**

** Fall 2019**

**Fall 2021**

**CSE208L Object oriented programming**

Submitted by: **Ashfaq Ahmad**

Registration No. : **19PWCSE1795**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Sumayyea salahuddin**

January 14, 2021

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

.

**Objectives of the lab:**

1 Understand the difference between a regular function and a friend function

2 Explain the concept of friend function.

3 Develop a friend function.

4 Explain the concept of friend class.

5 Develop a friend class.

**Activity no 1**

Create a class RationalNumber that stores a fraction in its original form (i.e. without finding the equivalent floating pointing result). This class models a fraction by using two data members: an integer for numerator and an integer for denominator………………………………………………………………………..

**Source code:**

#include<iostream>

using namespace std;

class RationalNumber {

private:

int top, bottom;

public:

RationalNumber() {

top = 0;

bottom = 1;

}

RationalNumber(int n1, int n2) {

top = n1;

bottom = n2;

}

void showRN() {

cout<<top <<"/"<<bottom<<endl;

}

friend bool operator>=(RationalNumber, RationalNumber);

friend bool operator<=(RationalNumber, RationalNumber);

friend bool operator==(RationalNumber, RationalNumber);

friend RationalNumber operator+(RationalNumber, RationalNumber);

friend RationalNumber operator/(RationalNumber, RationalNumber);

friend bool operator>(RationalNumber, RationalNumber);

friend bool operator<(RationalNumber, RationalNumber);

friend RationalNumber operator-(RationalNumber, RationalNumber);

friend RationalNumber operator\*(RationalNumber, RationalNumber);

friend bool operator!=(RationalNumber, RationalNumber);

};

RationalNumber operator+(RationalNumber n1, RationalNumber n2) {

RationalNumber returnvalue(n1.top \* n2.bottom + n1.bottom \* n2.top,n1.bottom \* n2.bottom);

return returnvalue;

}

RationalNumber operator-(RationalNumber n1, RationalNumber n2) {

RationalNumber returnvalue(n1.top \* n2.bottom - n1.bottom \* n2.top,n1.bottom \* n2.bottom);

return returnvalue;

}

RationalNumber operator\*(RationalNumber n1, RationalNumber n2) {

RationalNumber returnvalue(n1.top \* n2.bottom, n1.bottom \* n2.bottom);

return returnvalue;

}

RationalNumber operator/(RationalNumber n1, RationalNumber n2) {

RationalNumber returnvalue(n1.top \* n2.bottom, n1.bottom \* n2.top);

return returnvalue;

}

bool operator>(RationalNumber n1, RationalNumber n2) {

float n3 = n1.top/n1.bottom, n4 = n2.top/n2.bottom;

if(n3>n4) return true;

return false;

}

bool operator<(RationalNumber n1, RationalNumber n2) {

float n3 = n1.top / n1.bottom, n4 = n2.top/n2.bottom;

if(n3 < n4) return true;

return false;

}

bool operator<=(RationalNumber n1, RationalNumber n2) {

float n3 = n1.top / n1.bottom, n4 = n2.top/n2.bottom;

if(n3 <= n4) return true;

return false;

}

bool operator>=(RationalNumber n1, RationalNumber n2) {

float n3 = n1.top / n1.bottom, n4 = n2.top/n2.bottom;

if(n3 >= n4) return true;

return false;

}

bool operator==(RationalNumber n1, RationalNumber n2) {

float n3 = n1.top / n1.bottom, n4 = n2.top/n2.bottom;

if(n3 == n4) return true;

return false;

}

bool operator!=(RationalNumber n1, RationalNumber n2) {

float n3 = n1.top / n1.bottom, n4 = n2.top/n2.bottom;

if(n3!= n4) return true;

return false;

}

int main() {

RationalNumber a(83, 11), b(3, 7);

cout<<"A + B is ";

RationalNumber n1 = a + b;

b = a + b;

b.showRN();

n1 = a - b;

n1.showRN();

n1 = a \* b;

n1.showRN();

n1 = a / b;

n1.showRN();

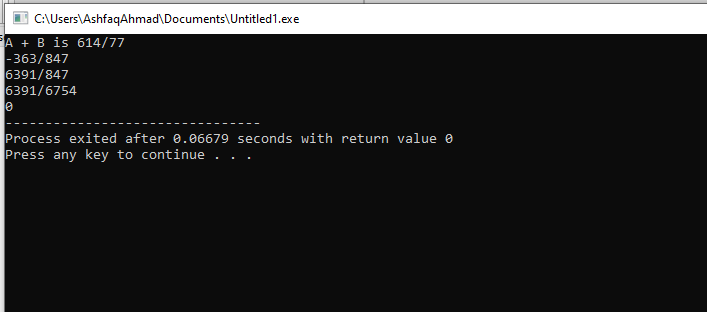
bool gr = a > b;

cout<<gr;

return 0;

}

**Compilation (testing and debugging):**



**Activity No 2**

Create a class called Time that has separate int member data for hours, minutes, and seconds…………………………………………………………………..

**Source code:**

#include<iostream>

using namespace std;

class Time {

private:

int s, m, h;

public:

Time(){

h = 0; m = 0; s = 0;

}

Time(int n1, int min, int sec) {

h = n1;

m = min;

s = sec;

}

void show() {

cout<<h<<":"<<m<<":"<<s<<endl;

}

friend Time operator+(Time, Time);

friend Time operator++(Time);

friend Time operator--(Time);

};

Time operator+(Time t1, Time t2) {

int hrs = t1.h + t2.h;

int min = t1.m + t2.m;

int sec = t1.s + t2.s;

if(sec > 59) {

sec = sec % 60;

min++;

}

if(min > 59) {

min = min % 60;

hrs++;

}

Time returnvalue(hrs, min, sec);

return returnvalue;

}

Time operator++(Time t) {

t.s++;

if(t.s > 59) {

t.m++;

t.s %= 60;

}

if(t.m > 59) {

t.h++;

t.m %= 60;

}

return t;

}

Time operator--(Time t) {

if(t.s == 0) {

if(t.m == 0) {

t.m = 59;

t.h--;

}else {

t.m--;

}

t.s = 59;

}else {

t.s--;

}

return t;

}

int main() {

Time t1(9, 12, 3), t2(1, 44, 13);

cout<<"T1 is ";

t1.show();

cout<<"T2 is ";

t2.show();

cout<<"T1 + T2 is ";

Time t3 = t1 + t2;

t3.show();

t1 = ++t1;

t1.show();

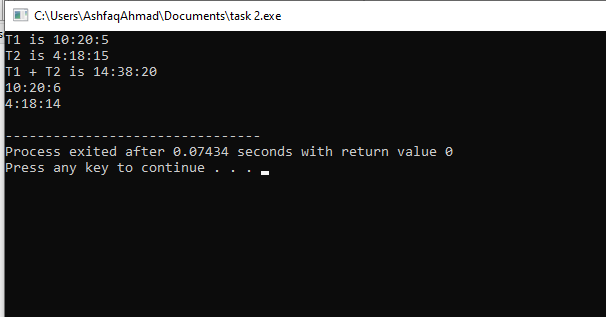
t2 = --t2;

t2.show();

return 0;

}

**Compilation:**



**THE END**

2nd code of task 2.

#include <iostream>

using namespace std;

class Time

{

int hour,minute,second;

public:

Time()

{

hour = 0;

minute = 0;

second = 0;

}

Time(int H,int M,int S)

{

if(H<0)

hour = 0;

else

hour = H;

if(M<0)

minute = 0;

else

minute = M;

if(S<0)

second = 0;

else

second = S;

}

void Show()

{

cout<<hour<<" : "<<minute<<" : "<<second<<endl;

}

friend void operator+(Time t1,Time t2);

friend void operator++(Time t1);

friend void operator--(Time t1);

};

void operator+(Time t1,Time t2)

{

Time t3;

t3.second = t1.second + t2.second;

t3.minute = t1.minute + t2.minute;

t3.hour = t1.hour + t2.hour;

if(t3.second >=60)

{

t3.second-=60;

t3.minute+=1;

}

if(t3.minute>=60)

{

t3.minute-=60;

t3.hour+=1;

}

if(t3.hour>24)

t3.hour-=24;

t3.Show();

}

void operator++(Time t)

{

t.second++;

if(t.second>=60)

{

t.second-=60;

t.minute+=1;

}

if(t.minute>=60)

{

t.minute-=60;

t.hour+=1;

}

if(t.hour>24)

t.hour-=24;

t.Show();

}

void operator--(Time t)

{

--t.second;

if(t.second<0)

{

t.second+=60;

t.minute-=1;

}

if(t.minute<0)

{

t.minute+=60;

t.hour-=1;

}

if(t.hour<=0)

t.hour+=24;

t.Show();

}

int main()

{

Time t1(30,59,59),t2(1,0,0),t3;

cout<<"First Time: ";

t1.Show();

cout<<"Second Time: ";

t2.Show();

cout<<"Sum of Times: ";

operator+(t1,t2);

cout<<"Post-Increment of First Time: ";

operator++(t1);

cout<<"Pre-Decrement of Second Time: ";

operator--(t2);

return 0;

}