

Лабораторна робота №3

Тема: Перевантаження операцій класу

Мета: Ознайомитись зі способами перевантаження операцій та навчитись використовувати їх при роботі з об'єктами

Завдання 1.

В класі `Int`, який розроблений в завданні №1 лабораторної роботи №1, перевизначте чотири цілочисельні арифметичні операції («+», «-», «*», «/») так, щоб їх можна було використовувати для операцій з об'єктами класу `Int`. Якщо результат будь-якої з операцій виходить за межі типу `int` (в 32-бітній системі), що може мати значення від 2 147 483 648 до -2 147 483 648, то операція повинна послати повідомлення про помилку і завершити програму. Такі типи даних корисні там, де помилки можуть бути викликані арифметичним переповненням, яке неприпустимо. Напишіть програму для перевірки цього класу.

Підказка: для полегшення перевірки переповнення виконуйте обчислення з використанням типу `long double`.

Код програми

```
#include <iostream>
#include <limits.h>
#include <cstdlib>
using namespace std;

class Int
{
private:
    int x;
public:

    Int(int x)
    {
        this->x=x;
    };
    Int()
    {this->x=0;}

    void SetInt(int x)
```

```

{
    this->x=x;
}
int GetInt()
{
    return x;
}

void Sum(Int n1, Int n2)
{
    x= n1.GetInt()+n2.GetInt() ;

}
void Null()
{
    x=0;
}

Int operator =(const Int &b)
{
    this->x = b.x;
    return *this;
}

Int operator +(const Int &b)
{
    long double x=(long double)this->x+b.x;

    if(x<INT_MIN || x>INT_MAX)
    {
        cout<<"Error";
        exit(0);
    }
    else {
        return Int((int)x);
    }
}

Int operator -(const Int &b) const
{

    long double x=(long double)this->x-b.x;

    if(x<INT_MIN || x>INT_MAX)
    {
        cout<<"Error";
        exit(0);
    }
    else {
        return Int((int)x);
    }
}

Int operator /(const Int &b)  const
{

    long double x=(long double)this->x/b.x;

```

```

        if(x<INT_MIN || x>INT_MAX)
        {
            cout<<"Error";
            exit(0);
        }
        else {
            return Int((int)x);
        }
    }

    Int operator *(Int &b) const
    {

        long double x=(long double)this->x*b.x;

        if(x<INT_MIN || x>INT_MAX)
        {
            cout<<"Error";
            exit(0);
        }
        else {
            return Int((int)x);
        }
    }

    friend ostream &operator<<(ostream &stream, Int &obj)
    {
        stream<<obj.x<<endl;
        return stream;
    }

};

int main()
{
    Int n1(21),n2(3),n3;
    cout<<"multiplication=";
    n3=n1*n2;
    cout<<n3;
    cout<<"division=";
    n3=n1/n2;
    cout<<n3;
    cout<<"subtraction=";
    n3=n1-n2;
    cout<<n3;
    cout<<"addition=";
    n3=n1+n2;
    cout<<n3;

    return 0;
}

```

Результат

```
C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe
multiplication=63
division=7
subtraction=18
addition=24
```

Завдання 2.

Для класу, який розроблено згідно індивідуального завдання лабораторної роботи № 2, визначити операції: - зчитування з потоку вводу `std::cin`; - виводу у потік `std::cout`. Перевірити роботу перевизначених функцій у функції `main()` за допомогою коду:
НазваКласуЗгідноВаріанта `myObject; std::cin >> myObject; std::cout << myObject;`

Код програми

```
#include <iostream>
#include<string.h>
using namespace std;

class Flat
{
    char *Size;
    int SIZE;
public:
    Flat()
    {
        this->SIZE=0;
        this->Size=nullptr;
    }

    Flat(char *_Size, int _SIZE)
    {
        this->Size=new char[30];
        strcpy(Size,_Size);
        this->SIZE=_SIZE;
    }

    Flat(const Flat &obj)
    {
        this->Size=new char[30];
        strcpy(this->Size,obj.Size);
        this->SIZE=obj.SIZE;
    }

    void SetSize(char * _Size)
```

```

{

    Size=nullptr;
    this->Size=new char[30];
    strcpy(Size,_Size);

}

char *GetSize()
{

    return this->Size;

}

void SetSIZE(int _SIZE)
{
    this->SIZE=0;
    this->SIZE=_SIZE;
}
int GetSIZE()
{
    return this->SIZE;
}
void Print() const
{
    cout<<"Char="<<this->Size<<"    Int="<<this->SIZE<<endl;
}
void Input()
{
    cout<<"Enter Int Size=";
    cin>>SIZE;
    this->Size=new char[30];
    cout<<"Enter Char *Size=";
    cin>>Size;

}
~Flat()
{
    if(this->Size)
    {delete this->Size;}
    cout<<"Object is delete"<<endl;
}

friend ostream & operator <<(ostream &stream,Flat &obj)
{
    stream<<"Char *="<<obj.Size<<"    "<<"Int="<<obj.SIZE<<endl;
    return stream;
}

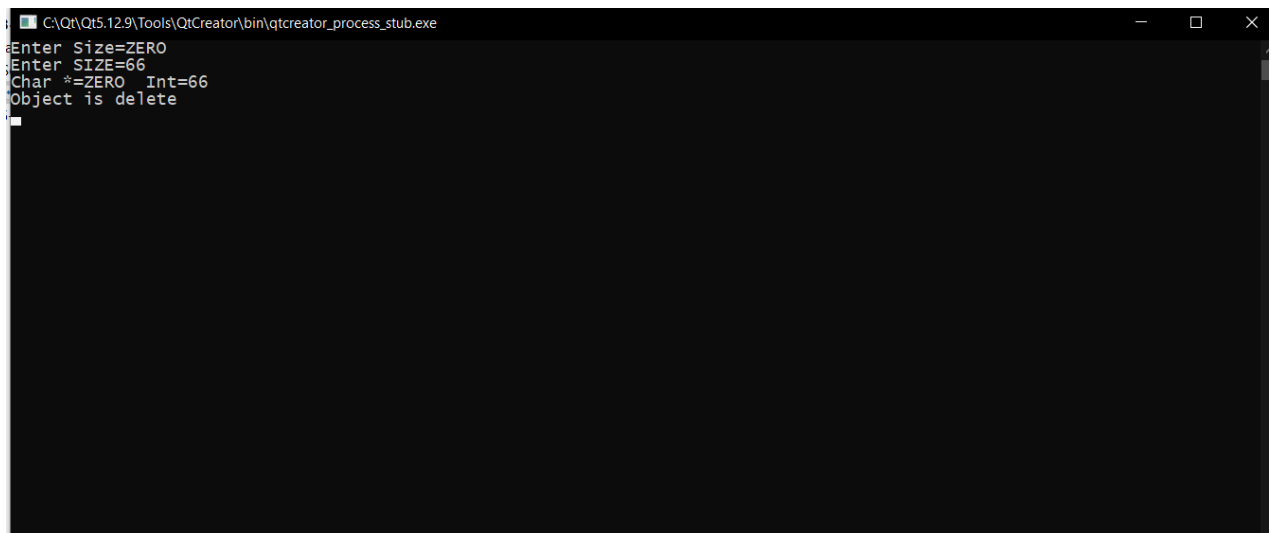
friend istream& operator >>(istream& is, Flat& x)
{
    x.Size=new char[30];
    cout<<"Enter Size=";
    is>>x.Size;
    cout<<"Enter SIZE=";
    is>>x.SIZE;
    return is;
}

```

```
};
```

```
int main()  
{  
    Flat n;  
    cin>>n;  
    cout<<n;  
    return 0;  
}
```

Результат

A screenshot of a Qt Creator console window. The title bar shows the path 'C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe'. The console output is as follows:
Enter Size=ZERO
Enter SIZE=66
Char *=ZERO Int=66
Object is delete
The text is displayed in a monospaced font on a dark background.

Завдання 3.

Для заданого варіанта індивідуального завдання виконати перевантаження операцій для зручності роботи з об'єктами. При необхідності оголосить певні операторні функції друзями класу.

ВАРІАНТ 11

Створити клас – квадрат з полями у закритій частині: координати головної діагоналі. Визначити необхідні конструктори, методи доступу, деструктор. Перевантажити потокові операції введення і виведення,

операції + (збільшення головної діагоналі), – (зменшення головної діагоналі), порівняння за периметром < , > та == .

Код програми

```
#include <iostream>
#include <math.h>

using namespace std;

class kub
{
private:
    double ax, ay, bx, by, Size, newSize;
public:
    kub()
    {
        this->ax=0;
        this->ay=0;
        this->bx=0;
        this->by=0;
        this->Size=0;
    };

    void Start()
    {
        cout<<"ax";
        cin>>this->ax;
        cout<<"ay";
        cin>>this->ay;
        cout<<"Enter sSize";
        cin>>this->Size;
        this->bx=this->by=Size/(sqrt(2));
    }

    void getpoint()
    {
        cout<<"ax"<<ax<<"  ay"<<ay<<"  bx"<<bx<<"  by"<<by<<endl;
    }

    void NewSize()
    {
        cout<<"Enter Size=";
        cin>>newSize;
    }

    kub operator =(const kub& obj)
    {
        bx=obj.bx;
        by=obj.by;
        Size=obj.Size;
        return *this;
    }

    kub operator +(const kub& obj) const
    {
```

```

        kub temp;
        temp.bx=(obj.ax+(obj.Size+obj.newSize)/(sqrt(2)));
        temp.by=(obj.ay+(obj.Size+obj.newSize)/(sqrt(2)));
        temp.Size=Size+newSize;
        return temp;
    }
    kub operator -(const kub& obj) const
    {
        kub temp;
        temp.bx=(obj.ax+(obj.Size-obj.newSize)/sqrt(2));
        temp.by=(obj.ay+(obj.Size-obj.newSize)/sqrt(2));
        temp.Size=Size+newSize;
        return temp;
    }

    kub operator >(const kub& obj) const
    {
        double P=4*((this->Size-this->newSize)/sqrt(2));
        if(P>obj.Size)
            cout<<"Correct"<<endl;
        else cout<<"Not correct"<<endl;
    }
    kub operator <(const kub& obj) const
    {
        double P=4*((this->Size-this->newSize)/sqrt(2));
        if(P<obj.Size)
            cout<<"Correct"<<endl;
        else cout<<"Not correct"<<endl;
    }
    kub operator ==(const kub& obj) const
    {
        double P=4*((this->Size-this->newSize)/sqrt(2));
        if(P=obj.Size)
            cout<<"Correct"<<endl;
        else cout<<"Not correct"<<endl;
    }
};
int main()
{
    kub k;
    k.Start();
    k.getpoint();

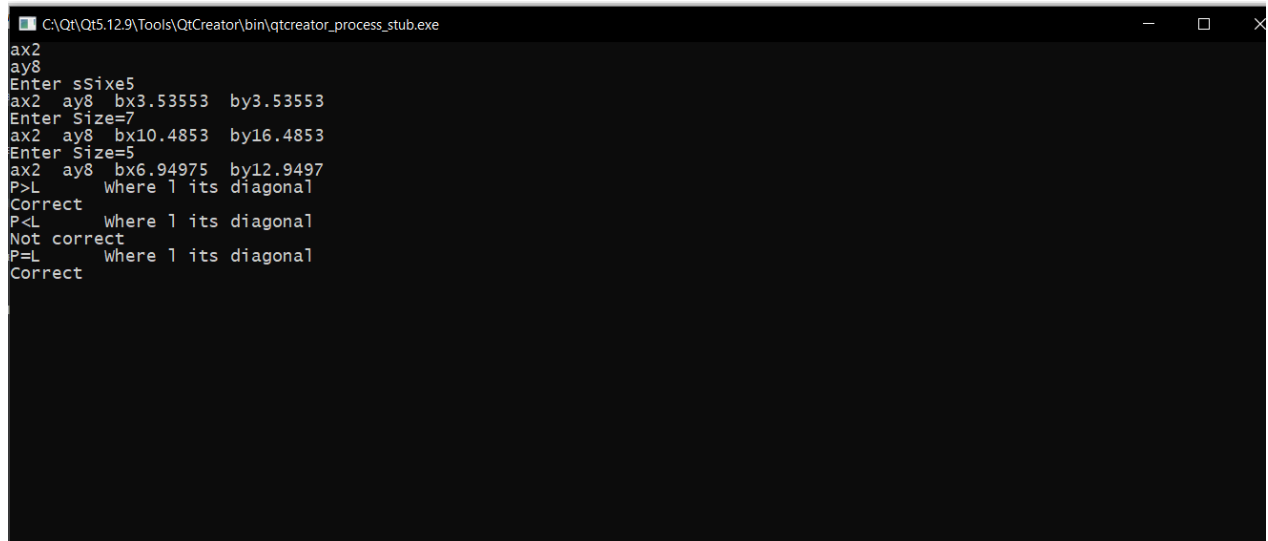
    k.NewSize();
    k=k+k;
    k.getpoint();

    k.NewSize();
    k=k-k;
    k.getpoint();
    cout<<"P>L\t Where l its diagonal"<<endl;
    k>k;
    cout<<"P<L\t Where l its diagonal"<<endl;
    k<k;
    cout<<"P=L\t Where l its diagonal"<<endl;
    k==k;
}

```



```
    return 0;  
}
```



```
C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe  
ax2  
ay8  
Enter sSize5  
ax2 ay8 bx3.53553 by3.53553  
Enter Size=7  
ax2 ay8 bx10.4853 by16.4853  
Enter Size=5  
ax2 ay8 bx6.94975 by12.9497  
P>L where 1 its diagonal  
Correct  
P<L where 1 its diagonal  
Not correct  
P=L where 1 its diagonal  
Correct
```

Висновок:

Ознайомитись зі способами перевантаження операцій та навчитись використовувати їх при роботі з об'єктами