

Лабораторна робота №9

Тема: Вивчення механізмів оброблення виняткових ситуацій.

Мета: навчитись обробляти ситуації появлення виняткових ситуацій, вивчити механізми їх оброблення.

Завдання 1

Додайте клас винятків до програми ARROVER таким чином, щоб індекси, що виходять за межі масиву, викликали генерацію винятку. Блокпастка catch може виводити користувачеві повідомлення про помилку

Код програми

```
////////program ARROVER////////
#include <iostream>
#include <string>
using namespace std;
#include <process.h> //for exit()
const int LIMIT = 100; //array size
////////////////////////////////////
class safearray
{
private:
    int arr[LIMIT];
public:
    int& operator [] (int n) throw (exception)
    {
        if( n< 0 || n>=LIMIT )
        {
            throw exception();
        }
        return arr[n];
    }
};
////////////////////////////////////
int main()
{
    int j;
    safearray sal;

    try
    {
        for(int j=0; j<200; j++)
        {
            sal[j] = j*10;
        }
    }
    catch (exception &ex)
    {
        cout<<ex.what()<<endl;
    }
}
```

```

try
{
    for(j=0; j<LIMIT; j++) //display elements
    {
        int temp = sal[j]; /*right* side of equal sign
        cout << "Element " << j << " is " << temp << endl;
    }
}
catch (exception &ex)
{
    cout<<ex.what()<<endl;
}

return 0;
}

```

Результат

```

C:\Qt\Qt5.12.9\tools\QtCreator\bin\qtcreator_process_stub.exe
std::exception
Element 0 is 0
Element 1 is 10
Element 2 is 20
Element 3 is 30
Element 4 is 40
Element 5 is 50
Element 6 is 60
Element 7 is 70
Element 8 is 80
Element 9 is 90
Element 10 is 100
Element 11 is 110
Element 12 is 120
Element 13 is 130
Element 14 is 140
Element 15 is 150
Element 16 is 160
Element 17 is 170
Element 18 is 180
Element 19 is 190
Element 20 is 200
Element 21 is 210
Element 22 is 220
Element 23 is 230
Element 24 is 240
Element 25 is 250
Element 26 is 260
Element 27 is 270
Element 28 is 280

```

Завдання 2

Модифікуйте програму з попереднього завдання таким чином, щоб в повідомлення про помилку входила інформація про значення індексу, який призвів до збою.

Код програми

```

////////program ARROVER////////
#include <iostream>
#include <string>
using namespace std;
#include <process.h> //for exit()
const int LIMIT = 100; //array size
////////////////////////////////////
class safearray
{
private:
    int arr[LIMIT];
public:
    string str;

```

```

int& operator [] (int n)
{
    if( n< 0 || n>=LIMIT )
    {
        str= to_string(n);
        throw exception();
    }
    return arr[n];
}
};
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

class myexception: public exception
{
    virtual const char* what() const throw()
    {
        return "My exception happened";
    }
} myex;
int main()
{
    int j,k;
    safearray sal;

    try
    {
        for(int j=0; j<200; j++)
        {
            k=j;
            sal[j] = j*10;
        }

    }
    catch (exception &ex)
    {
        cout<<ex.what()<<endl;
        cout<<"Index " <<k<<endl;
    }

    try
    {
        for(j=0; j<LIMIT; j++) //display elements
        {;
            int temp = sal[j]; /*right* side of equal sign
            cout << "Element " << j << " is " << temp << endl;
        }
    }
    catch (exception &ex)
    {
        cout<<ex.what()<<endl;
        cout<<"Index " <<k<<endl;
    }

    return 0;
}

```

Результат

```
C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe
Std::exception
Index 100
Element 0 is 0
Element 1 is 10
Element 2 is 20
Element 3 is 30
Element 4 is 40
Element 5 is 50
Element 6 is 60
Element 7 is 70
Element 8 is 80
Element 9 is 90
Element 10 is 100
Element 11 is 110
Element 12 is 120
Element 13 is 130
Element 14 is 140
Element 15 is 150
Element 16 is 160
Element 17 is 170
Element 18 is 180
Element 19 is 190
Element 20 is 200
Element 21 is 210
Element 22 is 220
Element 23 is 230
Element 24 is 240
Element 25 is 250
Element 26 is 260
```

Завдання 2

Модифікуйте текст програми, розробленої згідно індивідуального завдання лабораторної роботи №3, додавши до нього (в місця можливого виникнення помилок) процедури оброблення виняткових ситуацій, які будуть генерувати об'єкти класу, який міститиме такі атрибути: місце виникнення помилки; значення, яке призвело до помилки; параметризований конструктор; перевизначену операції виводу, яка виводитиме значення на екран (або зберігатиме виняткову ситуацію у файл).

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 Size=";
        cin>>this->Size;
        if(Size<0)
            throw exception();
        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;
    if(newSize<0)
        throw exception();
}

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;
point1:
    try {
        k.Start();
    }
    catch (exception &ex)
    {
        cout<<" The value is less than zero"<<endl;
        cout<<ex.what()<<endl;
        goto point1;
    }
    k.getpoint();
point2:
    try {

        k.NewSize();
    }
    catch (exception &ex )
    {
        cout<<" The value is less than zero"<<endl;
        cout<<ex.what()<<endl;
        goto point2;
    }
    k=k+k;
    k.getpoint();
point3:
    try {

        k.NewSize();
    }
    catch (exception &ex )
    {
        cout<<" The value is less than zero"<<endl;
        cout<<ex.what()<<endl;
        goto point3;
    }
    k=k-k;
    k.getpoint();
    cout<<"P>L\t Where 1 its diagonal"<<endl;
    k>k;
    cout<<"P<L\t Where 1 its diagonal"<<endl;
    k<k;
    cout<<"P=L\t Where 1 its diagonal"<<endl;
    k==k;

    return 0;
}

```

Результат

```
C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe
ax
0
ay0
Enter Size=-9
The value is less than zero
std::exception

ax0 ay0 bx0 by0
Enter Size=-9
The value is less than zero
std::exception

Enter Size=9
ax0 ay0 bx0 by0
Enter Size=-9
The value is less than zero
std::exception

Enter Size=8
ax0 ay0 bx-5.65685 by-5.65685
P>L where 1 its diagonal
Not correct
P<L where 1 its diagonal
Correct
P=L where 1 its diagonal
Correct
```

Висновок:

Навчився обробляти ситуації появлення виняткових ситуацій, вивчити механізми їх оброблення.