

# Обработка ошибок в C++

Exceptions

# Как сообщить об ошибке?

- Спецзначение.

```
FILE* file = fopen("Secret.txt", "rt" );  
if( file == NULL ) //Ааааа мы все умрем!!!
```

- Флаг и код ошибки

```
if( !WriteFile(hOut, Buffer, 256, NULL, NULL ) )  
{  
    cout << "Error: " << GetLastError();  
}
```

# Пичальки

- Плохие программисты не проверяют ошибки.
- Всегда тратим время на проверку, даже когда все хорошо.
- Функция вызывает функцию которая вызывает функцию...
- Иногда приходит Боромир

Нельзя просто так взять и вернуть ошибку...



```
class Matrix
```

```
{
```

```
...
```

```
Matrix operator* (const Matrix& right ) const {
```

```
if( this->Col != right.Row )
```

```
    return !?!?!?
```

```
...
```

```
}
```

```
...
```

```
};
```



# Пичальки

- Плохие программисты не проверяют ошибки.
- Всегда тратим время на проверку, даже когда все хорошо.
- Функция вызывает функцию которая вызывает функцию...
- Иногда приходит Боромир
- Конструкторы вообще ничего не возвращают

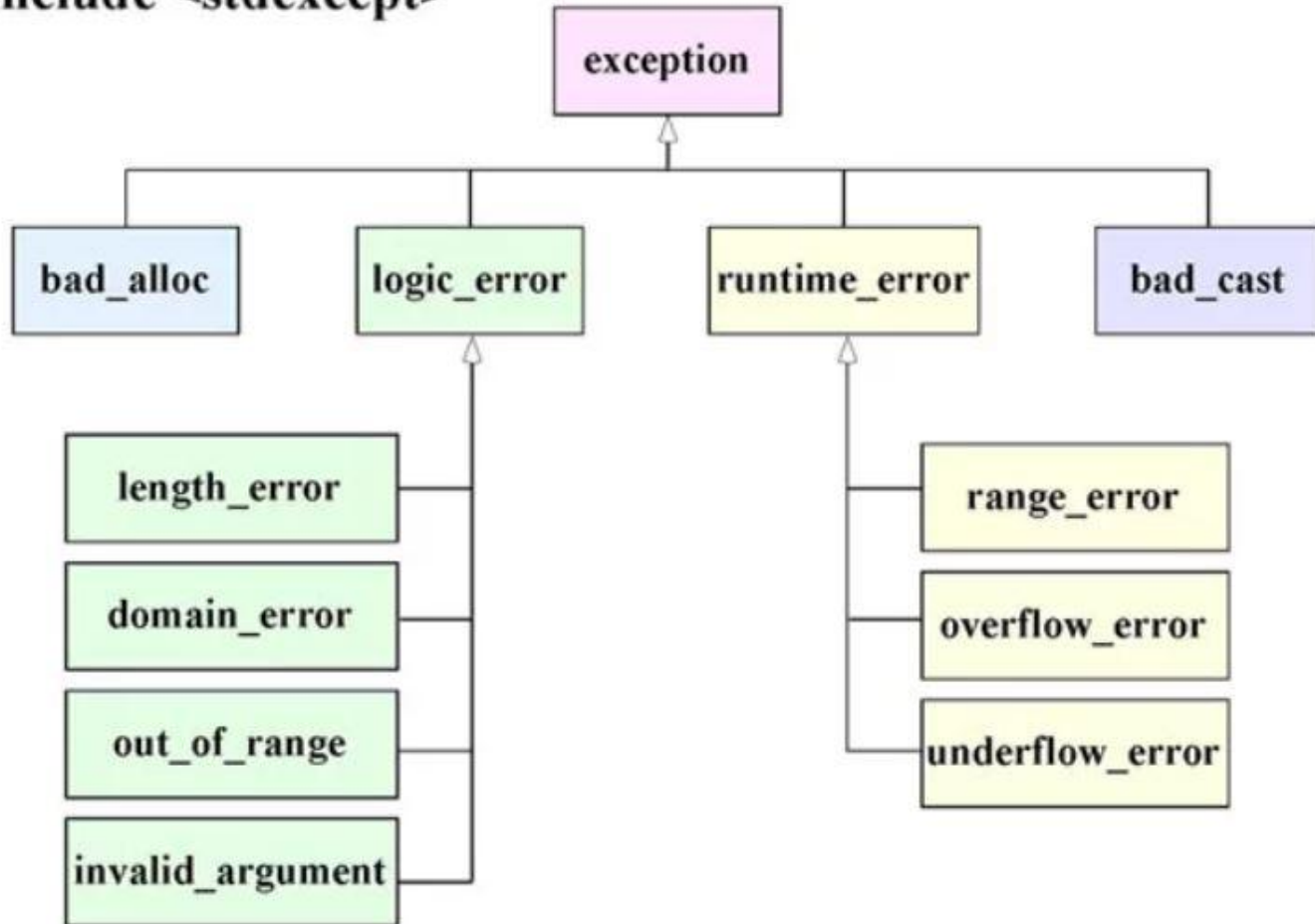
# Исключения (Exceptions)

- Обнаружил ошибку - кидайся ~~какаш~~ исключением, и ни о чем больше не думай.
- Хочешь обработать ошибку - лови и обрабатывай.
- Не поймал? Сам виноват.
- В ошибку можно засунуть много всякой информации.
- Не надо проверять каждую функцию, можно проверять целую кучу кода сразу.

# Exceptions

## C++ Exception Classes

```
#include <stdexcept>
```





# Создание своих классов ошибок

```
class MyException: public std::exception
{
    std::string m_msg;
public:
    MyException( const char* msg ) :m_msg( msg ) {}
    const char* what() const { return m_msg.c_str(); }
};
```

```
void function1()
{
    try {
        std::cout << "    Enter in function1" << std::endl;
        throw MyException ("Моя ошибка");
        std::vector<int> v(5); v.at(10) = 1;
    }
    catch(...) {
        std::cout << " Function1 terminated by      exception"
        << std::endl;
        throw;
    }
    std::cout << " Exit from function1" << std::endl;
}
```

```
void function2()
{
    try {
        std::cout << "Enter in function2" << std::endl;
        function1();
    }
    catch(...)
    {
        std::cout << "Function2 terminated by exception"
                << std::endl;

        throw;
    }
    std::cout << "Exit from function2" << std::endl;
}
```

```
int main ( int argc, char* argv[])
{
    try {
        function2();
    }
    catch( const MyException& ex ) {
        std::cout << "MyException: " << ex.what();
    }
    catch( const std::exception& ex ) {
        std::cout << "std::exception: " << ex.what();
    }
    catch( ... ) {
        std::cout << "Unknown exception!";
    }
    return 0;
}
```

Enter in function2

Enter in function1

Function1 terminated by exception

Function2 terminated by exception

MyException: Моя ошибочка!

```
void function1()
{
    try {
        std::cout << "    Enter in function1" << std::endl;
        //throw MyException ("Моя ошибка");
        std::vector<int> v(5); v.at(10) = 1; // std::out_of_range
    }
    catch(...) {
        std::cout << "    Function1 terminated by        exception"
        << std::endl;
        throw;
    }
    std::cout << "    Exit from function1" << std::endl;
}
```

Enter in function2

Enter in function1

Function1 terminated by exception

Function2 terminated by exception

std::exception: vector!

```
void function1()
{
    try {
        std::cout << "    Enter in function1" << std::endl;
        throw "Неведомая чушь";
        std::vector<int> v(5); v.at(10) = 1; // std::out_of_range
    }
    catch(...) {
        std::cout << " Function1 terminated by      exception"
        << std::endl;
        throw;
    }
    std::cout << " Exit from function1" << std::endl;
}
```



Enter in function2

Enter in function1

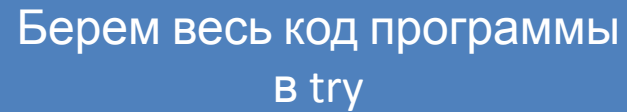
Function1 terminated by exception

Function2 terminated by exception

Unknown exception!

```
int main ( int argc, char* argv[])
{
    try {
        function2();
    }
    catch( const MyException& ex ) {
        std::cout << "MyException: " << ex.what();
    }
    catch( const std::exception& ex ) {
        std::cout << "std::exception: " << ex.what();
    }
    catch( ... ) {
        std::cout << "Unknown exception!";
    }
    return 0;
}
```

Берем весь код программы  
в try



```
int main ( int argc, char* argv[])
{
    try {
        function2();
    }
    catch( const MyException& ex ) {
        std::cout << "MyException: " << ex.what();
    }
    catch( const std::exception& ex ) {
        std::cout << "std::exception: " << ex.what();
    }
    catch( ... ) {
        std::cout << "Unknown exception!";
    }
    return 0;
}
```

Берем весь код программы  
в try

ЛОВИМ const ССЫЛКУ!

```
int main ( int argc, char* argv[])
{
    try {
        function2();
    }
    catch( const MyException& ex ) {
        std::cout << "MyException: " << ex.what();
    }
    catch( const std::exception& ex ) {
        std::cout << "std::exception: " << ex.what();
    }
    catch( ... ) {
        std::cout << "Unknown exception!";
    }
    return 0;
}
```

Берем весь код программы  
в try

ЛОВИМ const ССЫЛКУ!

Порядок ловушек важен!

```
int main ( int argc, char* argv[])
{
    try {
        function2();
    }
    catch( const MyException& ex ) {
        std::cout << "MyException: " << ex.what();
    }
    catch( const std::exception& ex ) {
        std::cout << "std::exception: " << ex.what();
    }
    catch( ... ) {
        std::cout << "Unknown exception!";
    }
    return 0;
}
```

Берем весь код программы  
в try

ЛОВИМ const ССЫЛКУ!

Порядок ловушек важен!

Ловушка "Ловись всё"  
должна стоять последней