C++语言程序设计

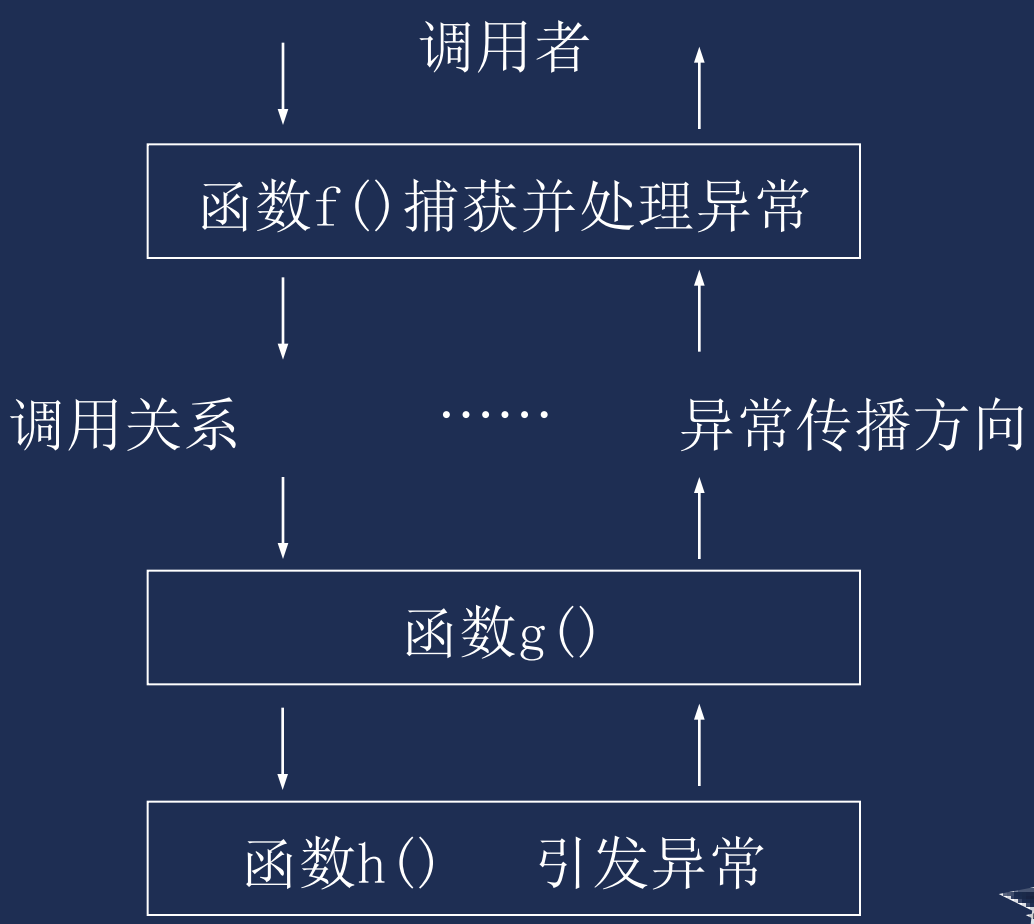
第十二章 异常处理

本章主要内容

- 异常处理的基本思想
- C++异常处理的实现
- 异常处理中的构造与析构
- 标准程序库异常处理
- 小结

异常处理的基本思想

异常处理的基本思想



C++异常处理的实现

异常处理的执行过程

- 抛掷异常的程序段

```
……
throw 表达式;
……
```

- 捕获并处理异常的程序段

```
try
    复合语句          保护段
catch (异常声明)
    复合语句          异常处理程序
catch (异常声明)
    复合语句
    …
catch (…)
    复合语句
```

异常处理的执行过程（续）

C++异常处理的实现

- 若有异常则通过throw操作创建一个异常对象并抛掷。
- 将可能抛出异常的程序段嵌在try块之中。控制通过正常的顺序执行到达try语句，然后执行try块内的保护段。
- 如果在保护段执行期间没有引起异常，那么跟在try块后的catch子句就不执行。程序从try块后跟随的最后一个catch子句后面的语句继续执行下去。
- catch子句按其在try块后出现的顺序被检查。匹配的catch子句将捕获并处理异常（或继续抛掷异常）。
- 如果匹配的处理器未找到，则运行库函数terminate将被自动调用，其缺省功能是调用abort终止程序。

例12-1处理除零异常


```
#include <iostream>
using namespace std;
int divide(int x, int y) {
    if (y == 0)
        throw x;
    return x / y;
}
int main() {
    try {
        cout << "5 / 2 = " << divide(5, 2) << endl;
        cout << "8 / 0 = " << divide(8, 0) << endl;
        cout << "7 / 1 = " << divide(7, 1) << endl;
    } catch (int e) {
        cout << e << " is divided by zero!" << endl;
    }
    cout << "That is ok." << endl;
    return 0;
}
```

程序运行结果如下：

```
5 / 2 = 2
8 is divided by zero!
That is ok.
```

异常接口声明

- 可以在函数的声明中列出这个函数可能抛掷的所有异常类型。
  - 例如：

```
void fun() throw(A, B, C, D);
```

- 若无异常接口声明，则此函数可以抛掷任何类型的异常。
- 不抛掷任何类型异常的函数声明如下：

```
void fun() throw();
```

异常处理中的构造与析构

- 找到一个匹配的catch异常处理后
  - 初始化异常参数。
  - 将从对应的try块开始到异常被抛掷处之间构造（且尚未析构）的所有对象进行自动析构。
  - 从最后一个catch处理之后开始恢复执行。
- 可用不带操作数的throw表达式在catch子句或catch子句内部调用的函数中将异常再次抛出

例12-2 异常处理时的析构

```
#include <iostream>
#include <string>
using namespace std;
class MyException {
public:
   MyException(const string &message) :
   message(message) {}
   ~MyException() {}
   const string &getMessage() const
   { return message; }
private:
   string message;
};
```

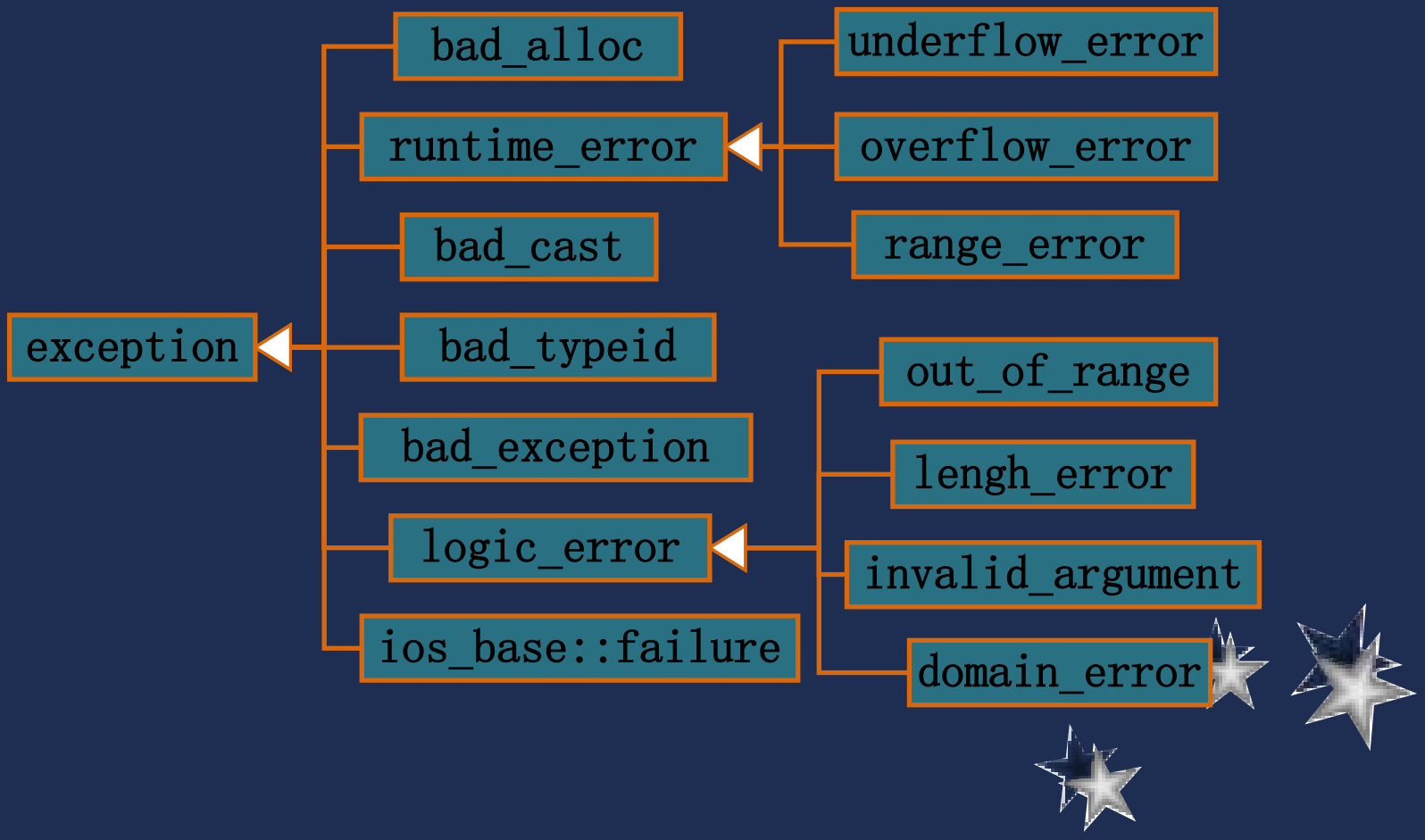
```
class Demo {
public:
  Demo() { cout << "Constructor of Demo" <<
  endl; }
  ~Demo() { cout << "Destructor of Demo" <<
  endl; }
};
void func() throw (MyException) {
  Demo d;
  cout << "Throw MyException in func()" <<
  endl;
  throw MyException("exception thrown by
  func()");
}
```

```
int main() {
  cout << "In main function" << endl;
  try {
     func();
  } catch (MyException& e) {
     cout << "Caught an exception: " <<
  e.getMessage() << endl;
  }
  cout << "Resume the execution of main()" <<
  endl;
  return 0;
}
```

程序运行时输出:

```
In main function
Constructor of Demo
Throw MyException in func()
Destructor of Demo
Caught an exception: exception thrown by func()
Resume the execution of main()
```

标准程序库异常处理


exception
bad_alloc
runtime_error
bad_cast
bad_typeid
bad_exception
logic_error
ios_base::failure
underflow_error
overflow_error
range_error
out_of_range
lengh_error
invalid_argument
domain_error

标准程序库的异常类

- exception：标准程序库异常类的公共基类
- logic_error表示可以在程序中被预先检测到的异常
  - 如果小心地编写程序，这类异常能够避免
- runtime_error表示难以被预先检测的异常

标准程序库的异常类

- logic_error和runtime_error两个类及其派生类都有一个接收const string&型参数的构造函数，传入具体错误信息
- 基类exception提供成员函数what()，用于返回错误信息
  - virtual const char *what() const throw();

例12-3：计算三角形面积

- 编写一个计算三角形面积的函数，函数的参数为三角形三边边长a、b、c，三角形面积用Heron公式计算。
- 在计算三角形面积的函数中需要判断输入的参数a、b、c是否构成一个三角形，若三个边长不能构成三角形，则需要抛出异常。

```
#include <iostream>
#include <cmath>
#include <stdexcept>
using namespace std;
```

```
double area(double a, double b, double c) throw
  (invalid_argument) {
  //判断三角形边长是否为正
  if (a <= 0 || b <= 0 || c <= 0)
    throw invalid_argument("the side length should
  be positive");
  //判断三边长是否满足三角不等式
  if (a + b <= c || b + c <= a || c + a <= b)
    throw invalid_argument("the side length should
  fit the triangle inequation");
  //由Heron公式计算三角形面积
  double s = (a + b + c) / 2;
  return sqrt(s * (s - a) * (s - b) * (s - c));
}
```

```
int main() {
  double a, b, c;      //三角形三边长
  cout << "Please input the side lengths of a
  triangle: ";
  cin >> a >> b >> c;
  try {
    double s = area(a, b, c);//尝试计算三角形面积
    cout << "Area: " << s << endl;
  } catch (exception &e) {
    cout << "Error: " << e.what() << endl;
  }
  return 0;
}
```

程序运行时输出1:

```
Please input the side lengths of a triangle: 3 4 5
Area: 6
```

程序运行时输出2:

```
Please input the side lengths of a triangle: 0 5 5
Error: the side length should be positive
```

程序运行时输出3:

```
Please input the side lengths of a triangle: 1 2 4
Error: the side length should fit the triangle inequation
```

小结与复习建议

- 主要内容
  - 异常处理的基本思想、C++异常处理的实现、异常处理中的构造与析构
- 达到的目标
  - 简单了解C++的异常处理机制