面向对象程序方法与C++程序设计

第7章　异常处理

大连理工大学
主讲人-于红

本章知识点



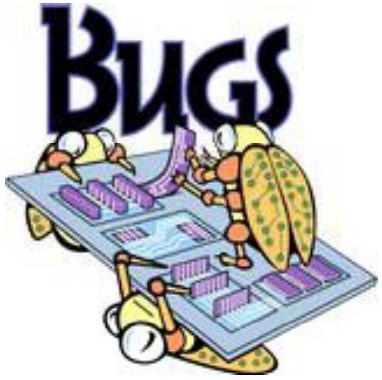
异常处理

- What 理解异常处理机制
- How 异常处理的实现
- When 异常的类型匹配

理解异常处理机制



- 异常指计算机硬件或软件中存在问题或错误。
  - 硬件异常
    - 设计错误，硬件部件老化失效等
  - 软件异常
    - 软件程序的错误或缺陷
      - ✓ 对各种流程分支考虑不全面
      - ✓ 对边界情况的处理不到位



理解异常处理机制



- 程序中出现的错误可以分：
  - 编译错误(Error)——一般是语法上存在的问题
  - 运行错误——指程序在运行过程中出现错误，简称异常(Exception)

- 异常的特点
  - 偶然性：程序运行中，异常并不总是会发生的。
  - 可预见性：异常的存在和出现是可以预见的。
  - 严重性： 一旦异常发生，程序可能终止，或者运行的结果不可预知。


```
int&  fun( )
{  int  temp =10;
   return temp;
}
```

```
int  t = fun( );
cout<<t<<endl;
```

理解异常处理机制



- 运行时错误的处理方法：
  - 出错时终止程序

```
void function(){
if (错误)  abort();
else  执行正常逻辑}
```

  - 用函数返回值作为错误标志

```
int function(){
  if (错误) return 1;
  else  { 执行正常逻辑；
     return 0;   }
}
```

  - 异常处理
    - 提供了处理程序运行时出现意外或异常情况的方法
    - 尝试可能未成功的操作、处理失败以及在事后清理资源

try – throw – catch

异常处理的实现



```
try
{ ……
   if(…) throw 类型m
    ……
}
catch(类型1 参数1)
{
   //针对类型1的异常处理
}
……
catch(类型n 参数n)
{
   //针对类型n的异常处理
}
```

try语句

包含可能产生错误的代码

throw语句

检测到错误后抛出异常

catch语句

将异常处理的语句放在其中，
异常被传递过来时就进行处理

异常处理的实现



【例1】 连续输入两个实数，程序将计算并输出这两个数相除的商。要注意除数不能为0。

```
double a=0; double b=0;
try{                                  //检测异常
     cout<<"请输入两个实数a和b： "<<endl;
      cin>>a>>b;
      if (b==0) throw b;       // 抛出异常
      cout<<"a/b="<<a/b<<endl;
}
catch(double)                         // 捕获异常
{  cout<<"除数不能为0。"<<endl; }
cout<<"完毕。"<<endl;
```

```
请输入两个实数：
1 2
a/b=0.5
完毕。
```

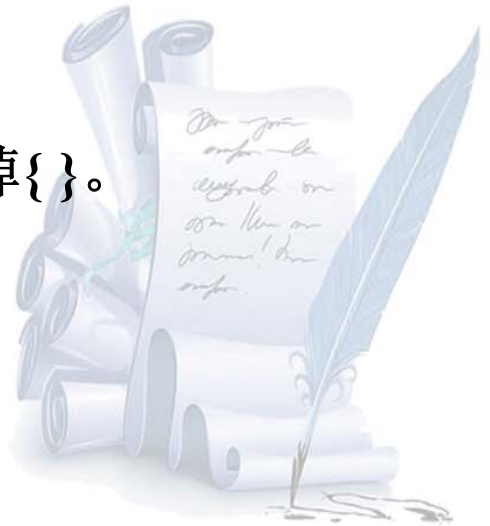
```
请输入两个实数：
1 0
除数不能为0。
完毕。
```

异常处理的实现



❖try-catch结构的说明：

- 被检测的语句或者函数的调用（可能出现异常的语句）必须放在try块中，否则不起作用。
- catch必须紧跟在try之后，在二者之间不能插入其他语句。
- try与catch块都是复合语句，即使块中只有一条语句也不能省略掉{ }。



异常处理的实现



【例2】 try-catch结构与throw也可不在同一个函数中。

```
double divide(double a, double b){
    if(b==0)  throw b;     // 抛出异常
   return a/b; }
void main(){
    double a=0; double b=0;
   try{                          //检测异常
     cout<<"请输入两个实数a和b： "<<endl;
       cin>>a>>b;  cout<<"a/b="<<divide(a,b)<<endl;
    }
   catch(double)              // 捕获异常
  { cout<<"除数不能为0。"<<endl; }
   cout<<"完毕。"<<endl;
}
```

该函数本身没有对异常进行处理，当有异常出现时，divide的执行终止。

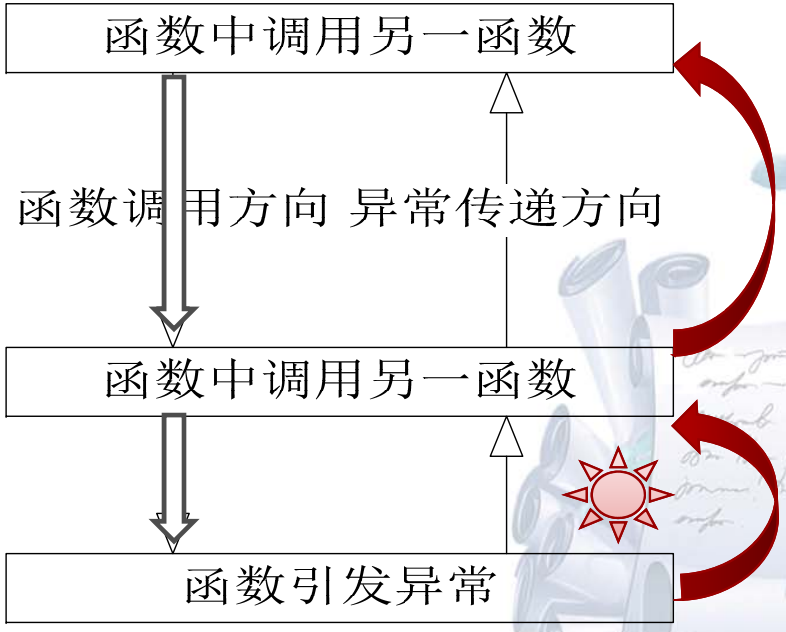
异常对象的类型匹配

异常处理的实现



异常的传递

- 异常的发生和处理不必在同一函数中
- 底层的函数可以着重解决具体逻辑问题
- 可以通过上层调用者完成异常的处理



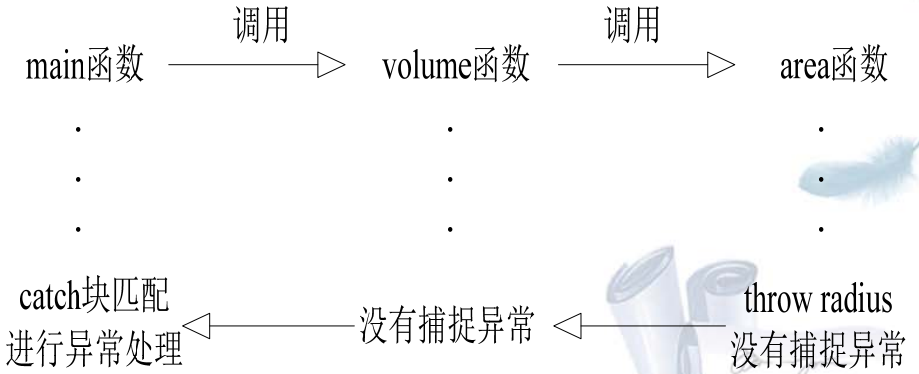
异常处理的实现



【例3】计算圆柱体体积的程序。

```
double area(double radius){
    if (radius<=0) throw radius;
    return 3.14*radius*radius;   }

double volume(double radius,double height){
     return area(radius)*height; }
void main(){
    double radius=0,height=0;
    cin>>radius>>height;
    try{ cout<<"该圆柱的体积是"<<volume(radius,hei
    }catch(double radius){
       cout<<"输入半径有误 radius="<<radius<<endl;
    } }
```



程序运行结果如下：

```
请输入圆柱的底面半径和高：
-3 9
输入半径有误 radius=-3
```

面向对象程序方法与C++程序设计

第7章　异常处理

大连理工大学
主讲人-于红