小　结

程序运行的错误是可以预料但是不能避免的，当程序运行出现异常的时候，我们应该能够保证程序不会因为错误而立刻退出，也就是说程序本身应该具有一定的识别和容错能力。在C++语言中，我们可以使用throw语句抛出异常对象，使用try-catch语句捕获异常对象，从而实现对异常的处理。当我们定义的异常处理语句不能够很好地处理当前的异常的时候，我们可以继续抛出该异常，交给上级函数进行处理，或者最终可以交给C++的内置异常类或者函数进行处理。

本节对C++标准程序库中标准异常类及功能进行了介绍，同时介绍了C++标准程序库对异常处理做的保证，即C++标准程序库在面对异常时，应该保证不会发生资源泄漏，也不能破坏容器的不变特性。

为了加强程序的可读性，使函数用户能够方便地知道所使用的函数会抛出哪些异常，可以在函数的声明中列出这个函数可能抛出的所有异常类型，这就是C++异常规范的相关问题。

习　题

一、填空题

1. 以下不属于异常处理可以解决的问题的是（　　）。

   A. 环境条件出现意外　　B. 用户操作不当

   C. 出现编译错误　　D. 除数为零

2. 以下关于异常处理的说法，正确的是（　　）。

   A. 异常的抛出和处理，必须在同一个函数中完成

   B. C++运行系统可以处理程序没有处理的异常

   C. 每个try语句只能和一个catch语句一起使用

   D. 异常不能被传播

3. 以下程序的运行结果为（　　）。

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

class Excp
{
public:
    Excp() {cout << "1";}
    Excp(Excp& e) {  cout<<"2";   }
    ~Excp() {cout << "3";    }
};

int divide(int a, int B.
{
    if( b==0 )
    {
```

```
        Excp e;
        throw e;
    }
    else
        return a/b;
}

int main()
{
    try{
        cout << divide(9,0) << endl;
    }catch (Excp e){}
    return 0;
}
```

A. 13　　B. 1233　　C. 122333　　D. 12223333

二、填空题

1. 下列程序段实现了一个简单的异常处理，用 try、throw 和 catch 检测除数为零的异常情况，请在横线处补充恰当的语句使程序具有完整的功能。

```
#include <iostream>
using namespace std;
int Divide(int x, int y){if(y==0) _____; 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(_____){cout<<"Exception by dividing zero.\n";}
   return 0;
}
```

2. 下面这个程序的输出结果为_____。

```
int main()
{
    for (int i=1; i<=5; i++){
       try{
          if (i == 3) throw ('o');
          if (i == 5) throw ("0");
          if (i == 1) throw (0);
       }
       catch(int) {cout << "int \n";}
       catch(char*) {cout << "char\n";}
       catch (…) {cout << "default \n";}
       cout << i<<endl;
    }
    return 0;
}
```

3. 下面这个程序的输出结果为_____。

```
void fun(int x)
```

```
{
    for (int i=1; i<=x; i++){
       try {
          if (i == 2) throw i;
          if (i ==3) throw 'i';
       }
       catch(int){cout << "One\n";}
       cout << "Two\n";
    }
}
void main()
{
    try {fun (2);
       fun (3);
    }
    catch(char) {cout << "Three\n";}
}
```

4. 下面的 throw 表达式哪些是错误的？为什么？对于合法的 throw 表达式，指出被抛出的异常类型。

(a) `class exceptionType(); throw exceptionType();`

(b) `int excpObj; throw excpobj;`

(c) `enum mathErr(overflow, underflow, zeroDivide);`

(d) `int * pi = &excpObj; throw pi;`

5. 如果函数有一个形式为 throw()的异常规范，那么它可以抛出什么异常？如果没有异常规范呢？