1. **解释error和exception的区别**：

Error：表示异常是比较严重的仅靠程序本身是不能解决的，程序一般不会运行。

Exception：为非致命异常，程序基本能够尽量处理。

1. 异常处理流程
2. **Try{}catch{}结构 finally用法**

try{

可能会出问题的程序语句； //一旦有异常出现，那么异常语句以后的程序语句都不会执行

}catch(Exception e){//处理所有异常的简单方法 直接使用父类exception 调用其中的方法printStackTrace()进行输出异常

e.printStackTrace()}

finally{//不管上面的程序有没有出现异常finally语句里的内容一样会执行

程序语句；

}

1. 捕获异常，抛出异常

利用catch（）{}来捕获异常 利用throws ， throw来抛出异常

1. 利用Exception处理所有异常
2. Throw和throws关键字 及其区别

.throw主要用于方法体之中，表示人为的抛出一个异常类的实例化对象。

.throws在方法的声明处使用，表示此方法不处理异常，所有异常交给方法调用处来处理。

1. 在调用处一定要处理异常 不能再往上抛
2. 使用throw表示人为的抛出一个异常，而throw一般都在方法中使用
3. **第五个代码模型：异常处理结构（核心）**

**package** importantprogram;

//第五个代码模型 try catch throws finally的组合使用

//程序要求：在MyMath类中定义方法div 此方法能在方法调用处抛出异常

// 在执行除法计算前输出一句 除法计算开始.....

//然后除法结束时输出一句 除法计算结束.....

**class** MyMath{

**public** **static** **int** div(**int** x,**int** y)**throws** Exception{

**int** result = 0;

**try**{

System.***out***.println("1.除法计算开始，打开计算机");

result = x/y;

}**catch**(Exception e){

e.printStackTrace();

}**finally**{

System.***out***.println("1.除法计算结束，关闭计算机");

}

**return** result;

}

}

**public** **class** TheFiveProgramMode {

**public** **static** **void** main(String args[]){

**try**{

System.***out***.println("除法计算结果：" + MyMath.*div*(10, 0));;

}**catch**(Exception e){

e.printStackTrace();

}

}

}

1. **Exception和runtimeException的区别。**

.runtimeException类也叫非致命异常类，当runtimeException及其子类所描述的异常发生后可以不通过try catch throws 捕获 抛出（即runtimeException 异常可以选择性进行处理），编译是可以通过的，知识在程序运行时交给JVM（Java虚拟机）来处理。

**多线程：**

将正在运行的程序成为进程。一个进程中包含多个线程，线程是进程中的实体。

**线程的创建方式：**

1. 通过继承thread类的方式。
2. 实现runnable接口。

Thread类的常用构造方法（4种）：

Runnable接口中只有run（）一个方法，实现runnable接口后必须覆写run方法。

实现runnable的类就可以成为线程，tread类就是因为实现了runnable类所以才具有了线程的功能。

**使用runnable接口和thread类的优缺点比较：**

Runnable接口更容易体现数据的共享。

Runnable接口可以实现多继承，然而thread类则不行。

要启动一个线程一定要调用thread类中的start方法。接口里的方法权限一定是public不管写不写都是。

表示时间日期的类型都用long表示。

主函数的线程优先级是5.

**线程中常用的方法：**

Start():

Run();

Sleep(int millsecond);

isAlive();

currentThread();

Interrupt();

**如何创建线程？如何启动线程？代码如何写？**

**线程的状态：** 创建 可执行 非可执行 消亡

**线程的控制：**

.线程的启动：

.线程的挂起（挂起情况）：

1. sleep（）方法
2. Join（）方法
3. Wait()与notify（）方法
4. Suspend（）与resume（）方法

请解释sleep（）和wait()的区别：

**线程的同步与死锁：**