1. 说明利用记事本和JDK开发执行一个Java程序的基本步骤。

* 新建：首先创建一个记事本，将保存的helloworld代码写入。
* 后缀：将文件后缀改为.java。
* Cmd：用windows+r 打开命令行，输入cmd。
* 位置：在打开界面输入cd+文件位置。
* 编译：然后输入 javac+文件名包括后缀。
* 运行：再输入java+文件名即可。

1. 包

Def：为了更好地组织类，Java 提供了包机制，用于区别类名的命名空间

建包方法

Package包名;

Ps：一个源文件只能有一个；包名应全部小写

作用

* 同类同包，便于查找（把功能相似或相关的类或接口组织在同一个包中，方便 类的查找和使用）
* 解决同名文件冲突
* 限定访问权限：拥有包访问权限的类才能访问某个包中的类

1. 标志符的命名规则

规则：

可以使用任意大小写字母、数字、\_ 和 $组成

其中数字不能打头，\_不能单独使用，不能有@%空格等特殊符号，不能 直接使用java关键字和保留字

习惯：名称要由意义的一个或多个单词组成，单词之间用下划线连接或大小 写区分。可使用一些约定俗成的方法，如驼峰命名方法。类和接口大驼 峰，即大写开头，推荐接口全部用 “I”做前缀；非静态变量和方法小驼 峰，即小写字母开头；常量全部大写

1. Java的访问控制符

访问修饰符置于类的成员定义前，限定对象对该类成员的访问权限

Private 仅该类持有

默认(不显示写出访问符) 类内、同个包

Protected 类内、同个包和子类可访问

Public 任意地方可以被访问

1. 域的隐藏

域指的是类中的成员变量

域的隐藏即子类定义了和父类同名的成员变量

子类访问父类通名变量的方法：

1. super关键字访问父类的同名变量
2. 调用父类中定义的getter方法，获取同名变量
3. 通过向上转型，获取父类中的同名变量
4. 覆盖：子类中，对和父类参数相同、返回类型相同的同名方法重新定义
5. 接口：一个只含抽象方法和全局变量的集合，使用interface关键字定义

声明：修饰符(public或没有) interface接口名字

1. 静态变量

Static关键字修饰

存储在方法区，方法区又叫静态区，包含整个程中永远序唯一的元素，如class、static变量

1. 多态

DEF：多态是同一个行为具有多个不同表现形式或形态的能力。多态就是同一个接口， 使用不同的实例而执行不同操作。

多态的形式：

编译时多态和运行时多态

1. 重载是编译时多态
2. 覆写分两种情况，当对象引用本类实例时是编译时多态；当对象引用(new)子类实例时是运行时多态
3. 实现接口是运行时多态

三个必要条件：继承、重写(覆盖)、父类引用指向子类对象

1. 异常

所有异常类都是Throwable的子类，他的两个重要子类是Error和Exception

Error：程序无法处理，程序员难以发现。

Exception分为RuntimeException编码时不检测编译时发现，如空指针异常、数组下标越界、算数异常、类型转换异常等， 和 CheckedException编码时就检测必须编译前改正，包括IO异常、SQL异常

异常处理：抛出异常+捕获异常

捕获异常部分，对于RuntimeException和Error允许忽略，CheckedException必须捕获或声明抛出。

* 捕获异常：

Try{

}catch(异常类型 ex){ //多次catch不能存在多个相同类型异常，最后的catch使用父类 Exception，确保无漏网之鱼

//PrintStackTrace();

}finally{

//勿写return，会导致之前的return失效

}

* 抛出异常：
  + Throws：声明可能要抛出的异常，谁调用谁处理
  + Throw：手动抛出异常，只能抛出Throwable及其子类的实例对象。自己跑自己处理：catch中处理；谁调用谁处理，可继续上抛抛出和throw对象相同的类型及其父类

1. AWT菜单
2. 组件
   1. MenuComponent菜单组件
   2. Component其他组件
      1. 基本组件类：按钮、文本框etc图形界面元素
      2. Container容器类：Component的子类，容纳其他组件
         1. Window不依赖其他容器独立存活
            1. 主界面Frame，带标题的框架窗口子类
            2. Dialog对话子类
         2. Panel长方形，必须存在在其他Container容器(Window和其子类中)
3. 设置布局管理器
4. AWT提供的事件监听模型的处理机制

事件源把自身可能发生的事件分别授予不同监听器处理

1. 程序、进程、线程

程序：是一个静态的概念，一般对应于操作系统中的一个可执行文件

进程：程序的一次执行周期，是资源分配的最小单位

线程：进程的一个子任务，是任务分配的最小单位、

1. 线程创建
   1. 继承Thread类，覆写run()，start()启动
   2. 实现Runnable接口，覆写run(),用start()启动
   3. 实现Callable接口， 覆写call()方法，⼀般是配合线程池工具 ExecutorService 来使用
2. 线程常用方法
   1. Sleep休眠：立马交出CPU，运行状态转为阻塞状态
   2. Yield让步：不立马交出CPU，不释放对象所，只给想过优先级的线程获取CPU的机会。运行状态转为就绪状态
   3. Join等待：当前线程等待别的线程执行完再恢复之乡，释放对象锁。运行状态转为阻塞状态
   4. Interrupted中断