**一.JDK JRE JVM概念**

**1.JDK**(Java Software Develop Kit)作为Java开发工具包，主要用于构建在Java平台上运行的应用程序、Applet 和组件等。

**2.JRE**(Java Runtime Environment，Java运行环境)，也就是Java平台。所有的Java程序都要在JRE下才能运行。JDK的工具也是Java程序，也需要JRE才能运行。为了保持JDK的独立性和完整性，在JDK的安装过程中，JRE也是安装的一部分。所以，在JDK的安装目录下有一个名为jre的目录，用于存放JRE文件。

**3.JVM**(Java Virtual Machine，Java虚拟机）是JRE的一部分（The JVM, which is the instance of the 'JRE' (Java Runtime Environment), comes into action when a Java program is executed. When execution is complete, this instance is garbage-collected）。它是一个虚构出来的计算机，是通过在实际的计算机上仿真模拟各种计算机功能来实现的。JVM有自己完善的硬件架构，如处理器、堆栈、寄存器等，还具有相应的指令系统。Java语言最重要的特点就是跨平台运行。使用JVM就是为了支持与操作系统无关，实现跨平台。

**二、java基本数据类型与数组**

1. 区分标识符与关键字

标识符是用来标识类名、变量名、方法名、类型名、数组名、文件名的有效字符序列。

2. 标识符命名规则

标识符由字母、下划线、美元符号和数字组成，长度不受限制

3. 基本数据类型

基本数据类型一共有八种

整数型：int long short byte

浮点型：float double

字符型： char

逻辑型：boolean

4. 类型转换运算

Java中数据的基本类型（不包括逻辑类型）按精度从“低”到“高”排列：  
       byte  short  char  int  long  float  double

5. 输入输出数据

使用Scanner类创建一个对象： Scanner reader=new Scanner(System.in);

6.输出基本型数据

使用System.out.println()或System.out.print()可输出串值、表达式的值

7.运算符、表达式和语句

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 运算符 | = | += | -= | \*= | /= | ++ | -- |
| 说明 | 赋值 | 加等于 | 减等于 | 乘等于 | 除等于 | 自增1 | 自减1 |

**三、JAVA类的继承与多态**

**1.多态后父类数据类型不能使用子类特有的属性和方法**

Object类是Java所有类的超类

equals方法:Object类中的equals方法用于检测一个对象与是否等于另一个对象

hashCode方法:获取散列码的方法

toString方法

**2. 多态的优点**

1. 消除类型之间的耦合关系

2. 可替换性

3. 可扩充性

4. 接口性

5. 灵活性

6. 简化性

**3.多态存在的三个必要条件:**

继承、重写

父类引用指向子类对象：Parent p = new Child();

**4. Java抽象类和内部类**

**类（class）：**

* 类是相似对象中共同属性和方法的集合体
* 在面向对象中定义类，就是在描述事物，就是在定义属性（变量）和行为（方法）。属性和行为共同成为类中的成员(成员变量和成员方法)。

**封装、继承和多态：**

* + 面向对象方法中软件设计的主体是类，类是相同属性和方法的封装体，因此类具有封装性
  + 子类可以在继承父类所有属性和方法的基础上，再增加自己特有的属性和方法，因此类具有继承性
  + 在一个类层次中，定义为根类的对象可被赋值为其任何子类的对象，并根据子类对象的不同而调用不同的方法，因此类具有多态性。

**类的这种封装性、继承性和多态性，是面向对象程序设计的三个最重要的特点。**

**类声明 ：**

格式:[<修饰符>] class<类名>[extends<父类名>] [implements<接口名表>]  
      {  
       类主体  
      }   
其中，class是定义类的关键字，当接口名多于一个时，用逗号分隔开。方括号表示该项是可选项。

**5.Java 多线程编程**

Java 给多线程编程提供了内置的支持。 一条线程指的是进程中一个单一顺序的控制流，一个进程中可以并发多个线程，每条线程并行执行不同的任务。

例 1

使用 JFrame 组件创建一个窗口，然后使用 JComboBox 类创建一个包含4个选项的下拉列表框。具体实现代码如下：

import javax.swing.JComboBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

public class JComboBoxDemo

{

public static void main(String[] args)

{

JFrame frame=new JFrame("Java下拉列表组件示例");

JPanel jp=new JPanel(); //创建面板

JLabel label1=new JLabel("证件类型："); //创建标签

JComboBox cmb=new JComboBox(); //创建JComboBox

cmb.addItem("--请选择--"); //向下拉列表中添加一项

cmb.addItem("身份证");

cmb.addItem("驾驶证");

cmb.addItem("军官证");

jp.add(label1);

jp.add(cmb);

frame.add(jp);

frame.setBounds(300,200,400,100);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

}