* JDK, JRE, JVM三者之间的关系，以及JDK、JRE包含的主要结构有哪些?

JDK = JRE + Java的开发工具（javac.exe, java.exe, javadoc.exe 等）

JRD = JVM + Java核心类库

* 简述对JVM, JRE, JDK的理解。

JVM 是 Java 虚拟机英文的缩写，JVM 是Java实现跨平台的基础，不同的操作系统具有不同的JVM，系统中有JVM Java程序才能够在系统中运行。

JRE 是 Java 运行环境的英文缩写，JRE 包含了 JVM 及 Java 运行时的核心类库，Java 程序如果需要能够在系统中运行必需要在系统中安装JRE。

JDK 是 Java 开发工具集的英文缩写，JDK 包含了 JRE 及 Java 开发工具。

它们之间的关系是 JDK > JRE >JVM

* 为什么要配置path环境变量？如何配置？

目的是为了在控制台的任何文件路径下，都可以调用jdk指定目录下的所有指令。

JAVA\_HOME = bin的上一层目录

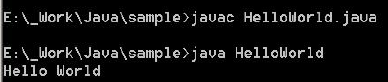
path = %JAVA\_HOME%\bin

* 简述环境变量path和classpath的作用？

配置环境变量的目的是为了让 Java 开发工具在任何路径下都能够被系统找到。

classpath 的作用是指定了 class 文件的路径信息。

* Java程序的编写、编译、运行步骤



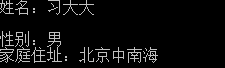
* Java程序编写的规则。
* 在配置环境、编译、运行各个步骤中常见的错误以及解决方法。
* 独立编写hello world程序，编译并运行。

|  |
| --- |
| public class HelloWorld{  public static void main(String[] args){  System.out.pringln("Hello World!");  }  } |

* 常用的几个命令行操作都有哪些？(至少4个)

|  |
| --- |
| cd  cd .  cd –  cd ..  md  rd  del  exit  dir  echo |

* 创建如下的类，使得运行的话可以输出：



|  |
| --- |
| package com.hfm;  /\*\*  \* 练习: 创建如下的类，使得运行的话可以输出  \* 姓名: 习大大  \* 性别: 男  \* 地址: 中南海  \*  \* @author hfm  \* @version 1.01 2020-03-18 19:05  \* @date 2020/3/18  \*/  public class PrintXiDaDa {  public static void main(String[] args) {  System.out.println("姓名: 习大大");  System.out.println("性别: 男");  System.out.println("地址: 中南海");  }  } |

* java语言的特点是什么？

面向对象性：两个基本概念：类、对象；三大特性：封装、继承、多态

健壮性：吸收了C/C++语言的优点，但去掉了其影响程序健壮性的部分（如指针、内存的申请与释放等），提供了一个相对安全的内存管理和访问机制

跨平台性：通过Java语言编写的应用程序在不同的系统平台上都可以运行。“Write once , Run Anywhere”

* java语言是否区分大小写?class关键字的作用,javac,java 的作用？

Java 语言区分大小写

class 文件是 Java 文件通过 Java 编译器 javac 编译之后的文件。

class 关键字的作用是指明一个类的关键字，用于标注 class 文件的开头。

javac : 启动 Java 编译器，将 Java 源文件编译成 class 文件。

java : 启动 JVM 解析并执行 class 文件。

* Java程序的编写、编译、运行步骤

1. 创建Java 源文件: 创建 .java 结尾的文本文件, 注意文件的扩展名不能隐藏
2. 编写 Class类: 创建Class 类文件名, 按照Class 类文件的格式编写
3. 编译 Java文件: 使用IDEA 等IDE 工具时会自动进行编译, 如果没有使用CMD 命令行模式进行编译, 即 javac Xxx.java 命令
4. 运行 Java 文件: 使用IDEA 等 IDE 工具可以通过图形化界面的方式运行Java 文件, 如果没有使用CMD 命令行方式运行 Class 文件, 即 java Xxx

* System.out.println()和System.out.print()什么区别呢？以下代码的运行效果是什么？

System.out.println();打印完后，会换行。

System.out.print();打印完后，不会换行。

* 简述main方法(是什么，什么特点，怎么用，何时用？)？

main 方法是程序的主入口，一个 Java 工程有且仅有一个 main 方法。

main 方法的格式是固定 public static void mian(String[] args){}

public 是四种访问权限修饰中权限最大的一个关键字，它表明 main 方法可以被任何类进行调用，如果 main 不用 public 进行修饰那么程序可能无法启动。

static 是静态的意思，它表明 main 方法是在对象出现之前就已经创建了，在类加载的时候就可以调用 main 方法。

void 表明 main 方法没有返回值，因为 main 方法一旦停止表明程序就停止了。

当一个程序需要一个人口时需要 main 方法。

* 一个".java"源文件中是否可以包括多个类（不是内部类）？有什么限制？

答：可以。但最多只有一个类名声明为public，与文件名相同。

* Java程序编写的规则?

|  |
| --- |
| // 类名首单词首字母需要大写, 其他单词都小写  // class 为关键字  class Demo{  // 一个Java 文件有且仅有一个main 函数, 他是程序运行的入口  // args 为启动程序的参数  public static void main(String [] args){  // 每行语句后需要加“；”进行结尾  System.out.println(“Hello World!”);  }  } |

* 判断是否正确

|  |
| --- |
| class Something {  public static void main(String[] something\_to\_do) {  System.out.println("Do something ...");  }  } |

答案: 正确。Java的class名字不必和其文件名相同。但public class的名字必须和文件名相同。

* 在配置环境、编译、运行各个步骤中常见的错误以及解决方法。

环境变量：需要配置JAVA\_HOME 环境变量，避免频繁修改相同环境变量， 以免对系统造成不可逆的破坏。

编译：编译的格式 javac 类名.java

运行：运行的格式 java 类名

* 为什么配置JAVA\_HOME

为了避免频繁修改path环境变量，如果不小心删除了path(系统自带的环境变量)，可能会导致系统部分功能不能使用。

path环境变量信息:

%SystemRoot%\system32;

%SystemRoot%;%SystemRoot%\System32\Wbem;

%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\

我们要尽量的避免去修改path的环境变量信息，因为path环境变量保存了系统自带的信息，如果一旦不小心删除失误，那么就会导致系统部分功能无法使用。创建一个新的环境变量，然后使用新的环境变量保存会发生变化的路径信息，然后在path环境变量中引用新的环境变量信息即可。D:\JDK\JDK\bin --> D:\JDK\JDK

注意:双百分号+ 环境变量的名称这个是引用了环境变量的信息。

* 为什么要配置JAVA\_HOME的环境变量信息？

为了避免频繁的修改path环境变量信息。有时候我们使用jdk对应版本的时候只会使用一次，然后就不再使用了，那么这时候我们可以通过set命令配置临时环境变量信息。

* 源文件名是否必须与类名相同？如果不是，那么什么情况下，必须相同？那么类名与源文件名可否不一致？

程序中若只有一个public修饰的类，且此类含main方法时，类名与文件名必须相同。

* JAVA CLASSPATH和PATH的区别

PATH: 用来指定java虚拟机(JVM)所在目录告诉java虚拟机,因为我们要常用到java给我们提供的一些工具对原代码进行一些处理或者

* Java的注释方式有哪几种，格式为何？

3种，分别为单行注释、多行注释、文档注释

单行注释：//

多行注释：/\* \*/

文档注释：/\*\* \*/

* GC是什么? 为什么要有GC?

答：GC是垃圾收集的意思（Gabage Collection）, 内存处理是编程人员容易出现问题的地方，忘记或者错误的内存回收会导致程序或系统的不稳定甚至崩溃，Java提供的GC功能可以自动监测对象是否超过作用域从而达到自动回收内存的目的，Java语言没有提供释放已分配内存的显示操作方法。

* 垃圾回收器的基本原理是什么？垃圾回收器可以马上回收内存吗？有什么办法主动通知虚拟机进行垃圾回收?

答：对于GC来说，当程序员创建对象时，GC就开始监控这个对象的地址、大小以及使用情况。通常，GC采用有向图的方式记录和管理堆(heap)中的所有对象。通过这种方式确定哪些对象是"可达的"，哪些对象是"不可达的"。

当GC确定一些对象为"不可达"时，GC就有责任回收这些内存空间。可以。程序员可以手动执行System.gc()，通知GC运行，但是Java语言规范并不保证GC一定会执行。

* 如果定义一个变量，如何使用变量?

对于成员变量来说，成员变量是有初始值的，byte short int long 的初始值为 0 ，float double 初始值为 0.0 ，boolean 数据类型初始值为 false ，char 数据类型初始值为 '0' ,引用数据类型的初始值为 null

对于局部变量来说，局部变量是没有初始值的必须要初始化才能够进行运算。

变量的初始化方式有两种：

静态初始化： int a = 10;

动态初始化： int a; a = 10;

* short s = 3; s = s+2; s+=2,有什么区别，为什么？

S = s+2 会报错，因为在运算时 s于2都会转化为int 数据类型，而 s 的数据类型没有发生变化，编译器将报告需要强制转换类型的错误Type mismatch: cannot convert from int to byte

S+=2 不会报错，由于 += 是 java 语言规定的运符，java 编译器会对它进行特殊处理，因此可以正确编译。

* 'a'+1，结果是什么？为什么？都做了什么事情呢？

结果是 98

char 数据类型在与 int 数据类型进行运算时会进行自动数据类型转换，'a' 对应的 ascll 码的数值为 97，自动数据类型转换后就是 97，97 + 1 = 98；

|  |
| --- |
| package com.edu.cjlu.hfm.homework;  public class Test {  public static void main(String[] args) {  System.out.println('a'+1);  }  } |

* ++在前，在后的区别？

++ 在前，先自加再运算

++ 在后，先运算再自加

* 将个人的基本信息（姓名、性别、籍贯、住址）打印到控制台上输出。各条信息分别占一行。

|  |
| --- |
|  |

* 结合\n(换行)，\t(制表符)，空格等在控制台打印出心形图案所示的效果。



|  |
| --- |
| package com.hfm;  /\*\*  \* 练习: 结合\n(换行)，\t(制表符)，空格等在控制台打印出心形图案所示的效果。  \*  \* @author hfm  \* @version 1.01 2020-03-18 19:01  \* @date 2020/3/18  \*/  public class PrintHeartShaped {  public static void main(String[] args) {  way1();  way2();  }  /\*\*  \* 打印心形方法 2  \*/  private static void way2() {  System.out.print("\t");  System.out.print("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.println("\*");  System.out.print("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("I love java");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.println("\*");  System.out.print("\t");  System.out.print("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.println("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.println("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.println("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\*");  System.out.print("\t");  System.out.print("\t");  System.out.println("\*");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print("\t");  System.out.print(" ");  System.out.print("\*");  }  /\*\*  \* 打印心形方法一  \*/  private static void way1() {  System.out.print("\t" + "\*" + "\t\t\t\t\t\t\t\t\t\t\t\t" + "\*" + "\t" + "\n");  System.out.print("\*" + "\t\t" + "\*" + "\t\t\t\t" + "I love Java" + "\t\t\t\t" + "\*" + "\t\t" + "\*" + "\n");  System.out.print("\t" + "\*" + "\t\t\t\t\t\t\t\t\t\t\t\t" + "\*" + "\t" + "\n");  System.out.print("\t\t" + "\*" + "\t\t\t\t\t\t\t\t\t\t" + "\*" + "\t\t" + "\n");  System.out.print("\t\t\t" + "\*" + "\t\t\t\t\t\t\t\t" + "\*" + "\t" + "\n");  System.out.print("\t\t\t\t" + "\*" + "\t\t\t\t\t\t" + "\*" + "" + "\t" + "\n");  System.out.print("\t\t\t\t\t" + "\*" + "\t\t\t\t" + "\*" + "" + "\t\t" + "\n");  System.out.print("\t\t\t\t\t\t" + "\*" + "\t\t" + "\*" + "" + "\t\t" + "\n");  System.out.print("\t\t\t\t\t\t\t" + "\*" + "\n");  }  } |

* 标识符的命名规则有哪些？

1. 标识符由 a-z A-Z 0-9 \_ $ 组成
2. 标识符不能以数字开头
3. 标识符长度没有限制，严格区分大小写
4. 不能使用保留字

不遵守，编译不通过。

* 练习：miles, Test, a++, --a, 4#R, $4, #44, apps, class, public, int, x, y, radius
* 标识符的命名规范有哪些？

1．包名: xxxyyyzzz 所有单词都小写

2. 类名、接口名：XxxYyyZzz 单词首字母大写

3. 变量名、方法名：xxxYyyZzz 首单词首字母小写，其他单词首字母大写

4. 常量名：XXX\_YYY\_ZZZ 所有字母大小，单词与单词之间通过下划线隔开

不遵守，编译运行都可以通过

* Java变量按照数据类型怎么划分？并指出Java的基本数据类型有哪8种，并指出各自占用的内存空间大小？

基本数据类型：4类8种

整形：byte short int long

浮点型：float double

字符型：char

布尔型：boolean

引用数据类型：类、接口、数组

* 说明基本数据类型变量之间自动类型提升的运算规则。

byte、short、char -> int -> long -> float -> double

* 说明基本数据类型变量之间强制类型转换的使用规则和强转可能出现的问题。

容量大 -> 容量小

使用强转符：()

数据溢出、精度损失。

* 基本数据类型有哪几类？包含String吗？

4类8种

整形 byte、sgort、int、long

浮点型 float double

布尔型 Boolean

字符型 char

不包含String型，String 型数据属于引用数据类型

* 写出基本数据类型自动转化的流程图

Byte、short、char -> int -> long -> float -> double

* 整型默认的是什么类型，浮点型（实数型）默认的是什么类型？

整形默认为 int 数据类型，浮点型默认为 double 数据类型

* 对于包名，类名接口名，变量名和函数名，常量名我们习惯如何格式来命名？

包名：所有单词小写，公司即组织机构网址逆序排放

接口名/类名：单词首字母大小

变量名/函数名：首单词首字母小写，其他单词首字母大小

常量名：所有单词都大写，单词与单词之间用下划线隔开

* 定义一个变量需要注意什么？

成员变量有初始值，可以不进行初始化就可以使用。局部变量一定要初始化之后才能够使用。

* 强制类型转化可能出现的问题

原数据大小超出强制类型转化之后的数据类型的数据范围，会造成数据溢出、精度损失。

* char型变量中能不能存贮一个中文汉字?为什么?

答：是能够定义成为一个中文的，因为java中以unicode编码，一个char占16个字节，所以放一个中文是没问题的

* 定义float f=3.4;是否正确? 表达式15/2\*2的值是多少

答:不正确。精度不准确,应该用强制类型转换，如下所示：float f=(float)3.4

答案：14

* String是最基本的数据类型吗

答：基本数据类型包括byte、int、char、long、float、double、boolean和short。

java.lang.String是java中定义的一个类，类都属于引用数据类型。

* 根据运算符的功能，我们把运算符分成哪几类？

算数运算符、逻辑运算符、位运算符、移位运算符、赋值运算符、比较运算符、三元运算符

* Java有没有goto

答：java中的保留字，现在没有在java中使用

* 用最有效的的方法算出2称以8等于几

答案：2<<3

* short s = 3; s = s + 2; s+=2,有什么区别，为什么？

short s = 3; s = s + 2; 会报错，因为存在强制数据类型转换而没有标出。

short s = 3； s+= 2; 不会报错，因为编译器会进行优化，预处理使其并不会报错。

* &和&&的区别?

相同点: 最后的运算结果都是相同的

不同点: && 的运算效率更高，一旦左边的为 false，右边不会在进行运算直接输出 false。

* 请写1个程序,这个程序要求用户从控制台输入两个数, 然后输出两个数的和.

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习请写1个程序,这个程序要求用户从控制台输入两个数, 然后输出两个数的和.  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class GetSum {  public static void main(String[] args) {  // 创建键盘输入对象  Scanner scanner = new Scanner(System.in);  System.out.println("请输入数字1");  int num1 = scanner.nextInt();  System.out.println("请输入数字1");  int num2 = scanner.nextInt();  int sum = num1 + num2;  System.out.println("数字1与数字2的和为" + sum);  }  } |

* 求用户输入矩形的长和宽,求出矩形的面积和周长,并将结果显示在屏幕上.

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 求用户输入矩形的长和宽,求出矩形的面积和周长,并将结果显示在屏幕上.  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class RectangularCalculation {  public static void main(String[] args) {  // 创建键盘输入对象  Scanner scanner = new Scanner(System.in);  System.out.println("请输入矩形从长");  int length = scanner.nextInt();  System.out.println("输入的长度为" + length);  System.out.println("请输入矩形的宽");  int width = scanner.nextInt();  System.out.println("输入的宽为" + width);  int perimeter = 2 \* length + 2 \* width;  System.out.println("矩形的周长为" + perimeter);  int area = length \* width;  System.out.println("矩形的面积为" + area);  }  } |

* 以下代码的运行结果

|  |
| --- |
| public static void main(String[] args) {  char x = 'x';  int i = 10;  System.out.println(true? x : i);  System.out.println(true? 'x' : 10);  } |

答案：

120

x

* 以下代码的执行结果

|  |
| --- |
| public static void main(String[] args) {  int a = 8, b = 3;  System.out.println(a>>>b);  System.out.println(a>>>b | 2); // 0B0001 | 0B0010 = 0B0011 = 3  } |

答案：

1

3

* 为抵抗洪水，战士连续作战89小时，编程计算共多少天零多少小时？

|  |
| --- |
| package com.hfm;  /\*\*  \* 练习: 为抵抗洪水，战士连续作战89小时，编程计算共多少天零多少小时？  \*  \* @author hfm  \* @version 1.01 2020-03-18 20:50  \* @date 2020/3/18  \*/  public class GetTime {  public static void main(String[] args) {  System.out.println("共计" + (89 / 24) + "天" + " " + (89 % 24) + "小时");  }  } |

* 今天是周二，100天以后是周几？

|  |
| --- |
| package com.hfm;  import java.util.Scanner;  /\*\*  \* 练习: 今天是周二，100天以后是周几？  \*  \* @author hfm  \* @version 1.01 2020-03-18 20:54  \* @date 2020/3/18  \*/  public class GetWeek {  public static void main(String[] args) {  System.out.println("请输入今天星期几(数字1-7)");  Scanner scanner = new Scanner(System.in);  int weekBegin = 0;  while (true) {  weekBegin = scanner.nextInt();  if (1 <= weekBegin && weekBegin <= 7) {  break;  } else {  System.out.println("请重新输入今天星期几(数字1-7)");  }  }  System.out.println("请输入经过的天数");  int day = scanner.nextInt();  int weekEnd = (day + weekBegin) % 7;  if (weekEnd == 0) {  weekEnd = 7;  }  System.out.println("今天是星期" + weekBegin + "经过" + day + "天后是星期" + weekEnd);  }  } |

* 将以下十进制数转换为十六进制和二进制123 256 87 62

将以下十六进制数转换为十进制和二进制0x123 0x25F 0x38 0x62

|  |
| --- |
| package com.hfm;  /\*\*  \* 练习: 将以下十进制数转换为十六进制和二进制 123 256 87 62  \* 将以下十六进制数转换为十进制和二进制 0x123 0x25F 0x38 0x62  \*  \* @author hfm  \* @version 1.01 2020-03-18 21:09  \* @date 2020/3/18  \*/  public class DecimalTest {  public static void main(String[] args) {  // 123 256 87 62 十进制数转换为二进制  System.out.println(Integer.toBinaryString(123));  System.out.println(Integer.toBinaryString(256));  System.out.println(Integer.toBinaryString(87));  System.out.println(Integer.toBinaryString(62));  // 123 256 87 62 十进制数转换为十六进制  System.out.println(Integer.toHexString(123));  System.out.println(Integer.toHexString(256));  System.out.println(Integer.toHexString(87));  System.out.println(Integer.toHexString(62));  // 0x123 0x25F 0x38 0x62 十六进制数转换为二进制  System.out.println(Integer.toBinaryString(0x123));  System.out.println(Integer.toBinaryString(0x25F));  System.out.println(Integer.toBinaryString(0x38));  System.out.println(Integer.toBinaryString(0x62));  // 0x123 0x25F 0x38 0x62 十六进制数转换为十进制  System.out.println(Integer.toString(0x123, 10));  System.out.println(Integer.toString(0x25F, 10));  System.out.println(Integer.toString(0x38, 10));  System.out.println(Integer.toString(0x62, 10));  }  } |

* 算术运算符的使用

|  |
| --- |
| class SuanShu{  public static void main(String[] args){  int a = 23;  int b = 12;  System.out.println(a + "+" + b + "=" + (a+b));  int sum = a + b;  System.out.println(a + "+" + b + "=" + sum);    int sub = a - b;  //System.out.println(a + "-" + b + "=" + a-b);//错误，原因是a + "-" + b + "=" + a的结果是字符串，字符串不能进行减法  System.out.println(a + "-" + b + "=" + (a-b));  System.out.println(a + "-" + b + "=" + sub);    int mul = a \* b;  System.out.println(a + "\*" + b + "=" + a\*b);  System.out.println(a + "\*" + b + "=" + mul);    //整数相除，只保留整数部分  int div = a / b;  System.out.println(a + "/" + b + "=" + a/b);  System.out.println(a + "/" + b + "=" + div);    double d = (double)a/b;//先把a的类型进行转换，转换成double类型，然后再和b相除  System.out.println(a + "/" + b + "=" + d);    int yu = a % b;  System.out.println(a + "%" + b + "=" + yu);    System.out.println("------------------特殊的取模----------------------");  System.out.println(5%2);//1  System.out.println(-5%-2);//-1  System.out.println(-5%2);//-1  System.out.println(5%-2);//1    System.out.println("------------------负号----------------------");  int num1 = 12;  int num2 = -num1;  System.out.println("num2=" + num2);    System.out.println("------------------自增----------------------");  int i = 0;  System.out.println("自增之前i=" + i);  i++;  System.out.println("自增第一次之后i=" + i);  ++i;  System.out.println("自增第二次之后i=" + i);  // 把i自增1，然后结果赋值给j，或者说，先算++i，然后再赋值  int j = ++i;  System.out.println("自增第三次之后i=" + i);  System.out.println("j="+j);  // 先算赋值，把i的值赋值给k，i原来是3，把3赋值给k，然后i在自增1，i变成4  int k = i++;  System.out.println("自增第四次之后i=" + i);  System.out.println("k="+k);  // 面试题：陷阱题  // 先赋值，把i原来的值重新赋值给i，不变，然后i自增，但是这个自增后的值没有在放回变量i的位置  i = i++;  System.out.println("自增第五次之后i=" + i);  }  }  自增之前i=0  自增第一次之后i=1  自增第二次之后i=2  自增第三次之后i=3  j=3  自增第四次之后i=4  k=3  自增第五次之后i=4 |

* 考查运算符的优先级, 写出输出的结果.

|  |
| --- |
| class Demo{  public static void main(String[] args){  int x=0,y=1;  if(++x==y-- & x++==1||--y==0)  System.out.println("x="+x+",y="+y);  else  System.out.println("y="+y+",x="+x);  }  } |

x = 2,y = 0;

* 编写代码实现两个变量值交换，int m = 3, n =5;

|  |
| --- |
| 答案一：  int temp = m;  m = n;  n = temp; |
| 答案二：  m = m + n;  n = m - n;  m = m - n; |
| 答案三：  m = m ^ n;  n = m ^ n;  m = m ^ n; |

* 如何求一个0~255范围内的整数的十六进制值，例如60的十六进制表示形式3C

|  |
| --- |
| //方式一：自动实现  String str1 = Integer.toBinaryString(60);  String str2 = Integer.toHexString(60); |
| //方式二：手动实现  int i1 = 60;  int i2 = i1&15;  String j = (i2 > 9)? (char)(i2-10 + 'A')+"" : i2+"";  int temp = i1 >>> 4;  i2 = temp & 15;  String k = (i2 > 9)? (char)(i2-10 + 'A')+"" : i2+"";  System.out.println(k+""+j); |

* 小明要到美国旅游，可是那里的温度是以华氏度为单位记录的。它需要一个程序将华氏温度（80度）转换为摄氏度，并以华氏度和摄氏度为单位分别显示该温度。

|  |
| --- |
| public static void main(String[] args) {  int h = 80;  double s = (h - 32) / 1.8;  System.out.println(h + "华氏度，是摄氏度" + Math.round(s));  } |

* 练习前++,后++,三元运算符, 写出输出结果

|  |
| --- |
| class Demo{  public static void main(String[] args){  int a=3,b=8;  int c=(a>b)?a++:b++;  System.out.println("a="+a+"\tb="+b+"\tc="+c); //  int d=(a>b)?++a:++b;  System.out.println("a="+a+"\tb="+b+"\td="+d); //  int e=(a<b)?a++:b++;  System.out.println("a="+a+"\tb="+b+"\te="+e); //  int f=(a<b)?++a:++b;  System.out.println("a="+a+"\tb="+b+"\tf="+f); //  }  } |

a=3 b=9 c=8

a=3 b=10 d=10

a=4 b=10 e=3

a=5 b=10 f=5

* 比较 + 与 +=

+ 号是算数运算符，有3个作用，表示正数、表示相加、用于String 类型的连接符

+= 是赋值表达式，只表示相加

* short s1 = 1; s1 = s1 + 1;有什么错? short s1 = 1; s1 += 1;有什么错

答：short s1 = 1; s1 = s1 + 1; （s1+1运算结果是int型，需要强制转换类型）

short s1 = 1; s1 += 1;（可以正确编译，编译器的常量优化）

35. 练习

|  |
| --- |
| String str1 = 4; //判断对错：no  String str2 = 3.5f + “”; //判断str2对错：yes  System.out.println(str2); //输出：”3.5”  System.out .println(3+4+“Hello!”); //输出：7Hello!  System.out.println(“Hello!”+3+4); //输出：Hello!34  System.out.println(‘a’+1+“Hello!”); //输出：98Hello!  System.out.println(“Hello”+‘a’+1); //输出：Helloa1 |

* 判断是否能通过编译

1）short s = 5;

s = s-2; //判断：no

2） byte b = 3;

b = b + 4; //判断：no

b = (byte)(b+4); //判断：yes

3）char c = ‘a’;

int i = 5;

float d = .314F;

double result = c+i+d; //判断：yes

4） byte b = 5;

short s = 3;

short t = s + b; //判断：no

39. 算术运算符：自加、自减

|  |
| --- |
| public class SignTest{  public static void main(String[] args){  int i1 = 10;  int i2 = 20;  int i = i1++;  System.out.print(“i=”+i);  System.out.println(“i1=”+i1);  i = ++i1;  System.out.print(“i=”+i);  System.out.println(“i1=”+i1);  i = i2--;  System.out.print(“i=”+i);  System.out.println(“i2=”+i2);  i = --i2;  System.out.print(“i=”+i);  System.out.println(“i2=”+i2);  }  } |

i=10 i=11

i=12 i=12

i=20 i2=19

i=18 i2=18

* 随意给出一个整数，打印显示它的个位数，十位数，百位数的值。

格式如下：

数字xxx的情况如下：

个位数：

十位数：

百位数：

例如：数字153的情况如下：

个位数：3

十位数：5

百位数：1

|  |
| --- |
| package com.hfm;  import java.util.Scanner;  /\*\*  \* 练习:随意给出一个整数，打印显示它的个位数，十位数，百位数的值。  \* 格式如下：数字xxx的情况如下:  \* 个位数：  \* 十位数：  \* 百位数：  \* 例如：数字153的情况如下：  \* 个位数：3  \* 十位数：5  \* 百位数：1  \*  \* @author hfm  \* @version 1.01 2020-03-18 21:41  \* @date 2020/3/18  \*/  public class PrintNum {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入一个整数");  int num = scanner.nextInt();  int hNum = 0;  int tNum = 0;  int sNum = 0;  if (num > 999) {  hNum = num % 1000 / 100;  tNum = num % 1000 % 100 / 10;  sNum = num % 1000 % 10;  } else {  hNum = num / 100;  tNum = num % 100 / 10;  sNum = num % 10;  }  System.out.println("数字" + num + "的情况如下:");  System.out.println("个位数:" + sNum);  System.out.println("十位数:" + tNum);  System.out.println("百位数:" + hNum);  }  } |

* short s = 3;s = s+2; ①s += 2; ② ②① 和 ② 有什么区别？

1. 报错，需要进行强制数据类型转化
2. 不会报错，编译器会进行优化

* 输出结果

|  |
| --- |
| int i = 1;  i \*= 0.1;  // 0  System.out.println(i);  i++;  // 1  System.out.println(i); |
| int m = 2;  int n = 3;  n \*= m++;  // 3  System.out.println("m=" + m);  // 6  System.out.println("n=" + n); |
| int n = 10;  n += (n++) + (++n);  // 31 + 1  System.out.println(n); |

* 区分好==和=的区别

|  |
| --- |
| boolean b1 = false;  if(b1 == true){  System.out.println("结果为真");  }else{  System.out.println("结果为假");  } |

= 号赋值运算符

== 号比较运算符

* 输出结果

|  |
| --- |
| int x = 1;  int y=1;  if(x++==2 & ++y==2){  x =7;  }  // x=2,y=2  System.out.println("x="+x+",y="+y); |
| int x = 1,y = 1;  if(x++==2 && ++y==2){  x =7;  }  // x=2,y=1  System.out.println("x="+x+",y="+y); |
| int x = 1,y = 1;  if(x++==1 | ++y==1){  x =7;  }  // x=7,y=2  System.out.println("x="+x+",y="+y); |
| int x = 1,y = 1;  if(x++==1 || ++y==1){  x =7;  }  // x=7,y=2  System.out.println("x="+x+",y="+y); |

* 输出结果

|  |
| --- |
| class Test {  public static void main (String [] args) {  boolean x = true;  boolean y = false;  short z = 42;  // if(y == true)  if ((z++ == 42) && (y = true)) {  z++;  }  if ((x = false) || (++z == 45)) {  z++;  }  // z= 46  System.out.println("z =" + z);  }  } |

* 使用三元运算符获取两个数中的较大数，获取三个数中的较大数

|  |
| --- |
| package com.hfm;  import java.util.Scanner;  /\*\*  \* 练习: 使用三元运算符获取两个数中的较大数，获取三个数中的较大数  \* @author hfm  \* @version 1.01 2020-03-18 22:39  \* @date 2020/3/18  \*/  public class GetMax {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入第一个数");  int num1 = scanner.nextInt();  System.out.println("请输入第二个数");  int num2 = scanner.nextInt();  System.out.println((num1>num2)? num1:num2);  System.out.println("请输入第三个数");  int num3 = scanner.nextInt();  System.out.println((num3>((num1>num2)? num1:num2))? num3:((num1>num2)? num1:num2));  }  } |

* 写出结果

|  |
| --- |
| class Demo{  public static void main(String[] args){  int m=0,n=3;  if(m>0){  if(n>2){  System.out.println("A");  }else{  System.out.println("B");  }  }  }  }  // answer:没有结果 |

* 实现对三个整数进行排序，输出时按照从小到大的顺序输出。

|  |
| --- |
| package com.hfm;  import java.util.Arrays;  /\*\*  \* 练习: 实现对三个整数进行排序，输出时按照从小到大的顺序输出。  \*  \* @author hfm  \* @version 1.01 2020-03-18 22:52  \* @date 2020/3/18  \*/  public class ThreeNumSort {  public static void main(String[] args) {  int[] arr = new int[]{232, 454, 29,34323,4545,4542,3235,232};  for (int i = 0; i < arr.length - 1; i++) {  for (int j = 0; j < arr.length - 1 - i; j++) {  int temp = 0;  if (arr[j] > arr[j + 1]) {  temp = arr[j];  arr[j] = arr[j + 1];  arr[j + 1] = temp;  }  }  }  System.out.println(Arrays.toString(arr));  }  } |

* 编写程序，从键盘接收整数参数。如果该数为1-7，打印对应的星期值，否则打印“非法参数”。

|  |
| --- |
| import java.util.Scanner;  class TestSwitch{  public static void main(String[] args){  Scanner input = new Scanner(System.in);  System.out.print("请输入星期值：");  int week = input.nextInt();  switch(week){  case 1:  System.out.println("星期一：Monday");  break;  case 2:  System.out.println("星期二：Tuesday");  break;  case 3:  System.out.println("星期三：Wednesday");  break;  case 4:  System.out.println("星期四：Thursday");  break;  case 5:  System.out.println("星期五：Friday");  break;  case 6:  System.out.println("星期六：Saturday");  break;  case 7:  System.out.println("星期天：Sunday");  break;  default:  System.out.println("非法星期值");  break;  }  }  } |

* 从键盘分别输入年、月、日，判断这一天是当年的第几天

|  |
| --- |
| import java.util.Scanner;  class TestDaysOfYear{  public static void main(String[] args){  Scanner input = new Scanner(System.in);  System.out.print("年：");  int year = input.nextInt();  System.out.print("月：");  int month = input.nextInt();  System.out.print("日：");  int day = input.nextInt();  int days = day;  //加前面几个月的满月天数  switch(month){  case 12:  //前面11个月的总天数  //days += 第11月的天数;  days += 30;  case 11:  //前面10个月的总天数  //days += 第10月的天数;  days += 31;  case 10:  days += 30;//九月  case 9:  days += 31;//八月  case 8:  days += 31;//七月  case 7:  days += 30;//六月  case 6:  days += 31;//五月  case 5:  days += 30;//四月  case 4:  days += 31;//三月  case 3:  days += 28;//二月  /\*if(闰年){  days++;  }  \*/  if(year % 4 ==0 && year % 100 != 0 || year%400==0){  days++;  }  case 2:  days += 31;//一月  }  System.out.println(year + "年" + month + "月" + day + "日是这一年的第" + days + "天");  /\*  int days = 0;  switch(month){  case 1:  days = day;  break;  case 2:  days = 31 + day;  break;  case 3:  //days = 31 + 二月的天数 + day;  days = 31 + 28 + day;  break;  case 4:  //days = 31 + 二月的天数 + 31 + day;  days = 31 + 28 + 31 + day;  break;  ....  }  if(闰年 && month >2){  days++;  }  \*/  }  } |

* switch是否能作用在byte上，是否能作用在long上，是否能作用在String上

答：switch（expr1）中，expr1是一个整数表达式。因此传递给 switch 和 case 语句的参数应该是

int、 short、 char 或者 byte。long不能作用于swtich.JDK1.7新加入了String类型。

* 要求用户输入两个数a和b，如果a能被b整除或者a加b大于1000，则输出a；否则输出b。

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:要求用户输入两个数a和b，如果a能被b整除或者a加b大于1000，则输出a；否则输出b。  \*  \* @author hfm  \* @version 1.01 2020-03-19 10:35  \* @date 2020/3/19  \*/  public class IfTest {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入正整数a");  int a = scanner.nextInt();  System.out.println("正整数a的值为" + a);  System.out.println("请输入正整数b");  int b = scanner.nextInt();  System.out.println("正整数a的值为" + b);  if (a % b == 0 || (a + b) > 1000) {  System.out.println(a);  } else {  System.out.println(b);  }  }  } |

* 使用条件结构实现，如果用户名等于字符‘青’，密码等于数字‘123’，就输出“欢迎你，青”，否则就输出“对不起，你不是青”。

提示：先声明两个变量，一个是char型的，用来存放用户名，一个是int型的，用来存放密码。

|  |
| --- |
| private static void load(Scanner scanner) {  System.out.println("请输入用户名");  String name = scanner.next();  System.out.println("请输入数字密码");  int password = scanner.nextInt();  if (("青").equals(name) && password == 123) {  System.out.println("welcome 青!");  } else {  System.out.println("用户名或密码错误!");  }  } |

* 编写程序：从键盘上读入一个学生成绩，存放在变量score中，根据score的值输出其对应的成绩等级：

score>=90 等级：A

70=<score<90 等级：B

60=<score<70 等级：C

score<60 等级：D

|  |
| --- |
| import java.util.Scanner;  class Exam1{  public static void main(String[] args){  Scanner input = new Scanner(System.in);  System.out.print("请输入学生的成绩：");  int score = input.nextInt();  if(score>=90){  System.out.println("等级：A");  }else if(score>=70){  System.out.println("等级：B");  }else if(score>=60){  System.out.println("等级：C");  }else{  System.out.println("等级：D");  }    /\*  if(score>=90){  System.out.println("等级：A");  }else if(score<90 && score>=70){  System.out.println("等级：B");  }else if(score<70 && score>=60){  System.out.println("等级：C");  }else{  System.out.println("等级：D");  }  \*/  }  } |

* 根据指定月份，打印该月份所属的季节。3,4,5 春季 6,7,8 夏季 9,10,11 秋季 12, 1, 2 冬季

|  |
| --- |
| import java.util.Scanner;  class TestSeason{  public static void main(String[] args){  Scanner input = new Scanner(System.in);  System.out.println("请输入当前月份：");  int month = input.nextInt();    switch(month){  case 3:  case 4:  case 5:  System.out.println("春季");  break;  case 6:  case 7:  case 8:  System.out.println("夏季");  break;  case 9:  case 10:  case 11:  System.out.println("秋季");  break;  case 12:  case 1:  case 2:  System.out.println("冬季");  break;  default:  System.out.println("非法月份");  }  }  } |

* 求ax2+bx+c=0方程的根。a,b,c分别为函数的参数，

如果a≠0，那么：

（1）当b2-4ac>0，则有两个解；

（2）当b2-4ac=0，则有一个解；

（3）当b2-4ac<0，则无解；

如果a=0,b≠0，那么，

提示1：Math.sqrt(num); sqrt指平方根

例如：求x2-4x+1=0方程的根

求x2-2x+1=0方程的根

|  |
| --- |
| import java.util.Scanner;  class Exer5{  public static void main(String[] args){  Scanner input = new Scanner(System.in);    System.out.println("一元二次方程：ax^2+bx+c=0");  System.out.print("请输入参数a：");  double a = input.nextDouble();    System.out.print("请输入参数b：");  double b = input.nextDouble();    System.out.print("请输入参数c：");  double c = input.nextDouble();  if(a!=0){  double temp = b\*b - 4\*a\*c;  if(temp==0){  double x = -b/(2\*a);  System.out.println("该方程是一元二次方法，有两个相同解：" + x);  }else if(temp>0){  double sqrt = Math.sqrt(temp);  double x1 = (-b+ sqrt)/(2\*a);  double x2 = (-b- sqrt)/(2\*a);  System.out.println("该方程是一元二次方法，两个不同解：" + x1 +"," + x2);  }else{  System.out.println("该方程是一元二次方法，在实数范围内无解！");  }  }else{  if(b!=0){  double x = -c/b;  System.out.println("该方程是一元一次方程，有一个解：" + x);  }else{  System.out.println("不是方程，是一个等式");  if(c == 0){  System.out.println("等式成立");  }else{  System.out.println("等式不成立");  }  }  }  }  } |

* 彩票游戏

假设你想开发一个玩彩票的游戏，程序随机地产生一个两位数的彩票，提示用户输入一个两位数，然后按照下面的规则判定用户是否能赢。

1)如果用户输入的数匹配彩票的实际顺序，奖金10 000美元。

2)如果用户输入的所有数字匹配彩票的所有数字，但顺序不一致，奖金 3 000美元。

3)如果用户输入的一个数字仅满足顺序情况下匹配彩票的一个数字，奖金1 000美元。

4)如果用户输入的一个数字仅满足非顺序情况下匹配彩票的一个数字，奖金500美元。

5)如果用户输入的数字没有匹配任何一个数字，则彩票作废。

提示：使用Math.random() 产生随机数

Math.random() 产生[0,1)范围的随机值

Math.random() \* 90：[0,90)

Math.random() \* 90 + 10：[10,100) 即得到 [10,99]

使用(int)(Math.random() \* 90 + 10)产生一个两位数的随机数。

|  |
| --- |
| import java.util.Scanner;  class TestCaiPiao{  public static void main(String[] args){  //1、随机产生一个两位数  //System.out.println(Math.random());//产生[0,1)  int number = (int)(Math.random()\*90 + 10);//得到[10,99]，即[10,100)  //System.out.println(number);    int numberShi = number/10;  int numberGe = number%10;    //2、用户输入一个两位数  Scanner input = new Scanner(System.in);  System.out.print("请输入一个两位数：");  int guess = input.nextInt();    int guessShi = guess/10;  int guessGe = guess%10;    if(number == guess){  System.out.println("奖金10 000美元");  }else if(numberShi == guessGe && numberGe == guessShi){  System.out.println("奖金3 000美元");  }else if(numberShi==guessShi || numberGe == guessGe){  System.out.println("奖金1 000美元");  }else if(numberShi==guessGe || numberGe == guessShi){  System.out.println("奖金500美元");  }else{  System.out.println("没中奖");  }    System.out.println("中奖号码是：" + number);  }  } |

* 赌数游戏

提供三个1-6的随机数，作为掷骰子得到的点数。如果各个点数相同，则为豹子。

如果三个骰子点数和，小于或等于9，则为“小”。

如果三个骰子点数和，大于9，则为“大”。

用户随机选择：押大、押小或者押豹子。通过判断，输出客户是否押正确。

|  |
| --- |
| import java.util.Scanner;  class GuessNumber{  public static void main(String[] args){    //1、产生一个[1-6]的数  int a = (int) (Math.random() \* 6 + 1);  //System.out.println(a);  int b = (int) (Math.random() \* 6 + 1);  //System.out.println(b);  int c = (int) (Math.random() \* 6 + 1);  //System.out.println(c);    /\*  a = 1;  b = 1;  c = 1;  \*/  String result = "";  if (a == b && b == c) {  result = "豹子";  } else if (a + b + c <= 9) {  result = "小";  } else {  result = "大";  }  //2、用户输入猜的结果  Scanner input = new Scanner(System.in);  System.out.print("请押宝，买定离手：(选择：大、小、豹子)");  String guess = input.next();  //3、判断结果  if (result.equals(guess)) {  System.out.println("猜对了");  } else {  System.out.println("猜错了");  }  }  } |

* 生肖问题

编写一个程序，为一个给定的年份找出其对应的中国生肖。

中国的生肖基于12年一个周期，每年用一个动物代表：rat（鼠）、ox（牛）、tiger（虎）、rabbit（兔）、dragon（龙）、snake（蛇）、horse（马）、sheep（羊）、monkey（候）、rooster（鸡）、dog（狗）、pig（猪）。

提示：2019年：猪 2019 % 12 == 3

|  |
| --- |
| import java.util.Scanner;  class ShengXiaoTest{  public static void main(String[] args){  Scanner input = new Scanner(System.in);  System.out.print("请输入年份：");  int year = input.nextInt();    switch(year%12){  case 0:  System.out.println("猴年");  break;  case 1:  System.out.println("鸡年");  break;  case 2:  System.out.println("狗年");  break;  case 3:  System.out.println("猪年");  break;  case 4:  System.out.println("鼠年");  break;  case 5:  System.out.println("牛年");  break;  case 6:  System.out.println("虎年");  break;  case 7:  System.out.println("兔年");  break;  case 8:  System.out.println("龙年");  break;  case 9:  System.out.println("蛇年");  break;  case 10:  System.out.println("马年");  break;  case 11:  System.out.println("羊年");  break;    }  }  } |

* switch是否能作用在byte上，是否能作用在long上，是否能作用在String上？

答案一：switch可以作用在byte上，不能作用在long上，JDK1.7之后可以作用在String上。

答案二：switch支持的类型byte,short,int,char，JDK1.5之后支持枚举，JDK1.7之后支持String类型。

* 下面程序片段的输出结果是？

|  |
| --- |
| public static void main(String[] args) {  int a = 3;  int b = 1;  if(a = b){  System.out.println("Equal");  }else{  System.out.println("Not Equal");  }  } |

答案：编译不通过

* What is the result when you compile and run the following code?

|  |
| --- |
| public class Test{  public void method(){  for(int i = 0; i < 3; i++) {  System.out.print(i);  }  System.out.print(i);  }  } |

A. 0122

B. 0123

C. compile error

D. none of these

答案：C

* 打印1-100之间13的倍数，使用for循环

|  |
| --- |
| package com.hfm.baisc.java;  /\*\*  \* 练习:打印1-100之间13的倍数，使用for循环  \*  \* @author hfm  \* @version 1.01 2020-03-19 11:00  \* @date 2020/3/19  \*/  public class Print13Divisible {  public static void main(String[] args) {  for (int i = 1; i <= 100; i++) {  if (i % 13 == 0) {  System.out.println(i);  }  }  }  } |

* 使用双重循环打印20 \* 8的矩形，使用for循环实现

|  |
| --- |
| for (int i = 0; i < 8; i++) {  for (int j = 0; j < 20; j++) {  System.out.print("\* ");  }  System.out.println();  } |

* 用for循环计算1000以内偶数的和

|  |
| --- |
| private static void getSum() {  int sum = 0;  for (int i = 1; i <= 1000; i++) {  if (i % 2 == 0) {  sum += i;  }  }  System.out.println(sum);  } |

* 执行如下代码后，c的值是多少？

|  |
| --- |
| public static void main(String[] args) {  int a = 0;  int c = 0;  do {  --c;  a = a - 1;  } while (a >= 0);  System.out.println("c = " + c);  } |

答案：c = -1

* 以下代码的运行结果？

|  |
| --- |
| public static void main(String[] args) {  int i=10;  while(i>0){  i = i +1;  if(i==10){  break;  }  }  System.out.println("i=" + i);  } |

答案一：是一个负数，因为i一直累加会超过int的存储范围

答案二：死循环

* 修正如下代码

下面是一段程序，目的是输出10个=，但是不小心代码写错了，现在需要修改代码，使得程序完成功能，但是只能“增加”或“修改”其中“一个”字符，很明显，将i--改为i++，可以完成功能，但是需要修改“两个”字符，所以并不是一个正确的答案？

|  |
| --- |
| public static void main(String[] args) {  int n=10;  for (int i = 0; i < n; i--) {  System.out.println("=");  }  } |

i<n修改为-i<n

* 输入长和宽，输出长方形，如：输入4和3, 将输出如下图形

####

####

####

|  |
| --- |
| private static void print2() {  for (int i = 0; i < 3; i++) {  for (int j = 0; j < 4; j++) {  System.out.print("#");  }  System.out.println();  }  } |

* 输入高度，输出直角三角形。如：输入4, 将输出如下图形

#

##

###

####

|  |
| --- |
| private static void print3() {  for (int i = 0; i < 4; i++) {  for (int j = 0; j <= i; j++) {  System.out.print("#");  }  System.out.println();  }  } |

* 输入高度，输出倒直角三角形。如：输入4, 将输出如下图形

####

###

##

#

|  |
| --- |
| private static void print4() {  for (int i = 0; i < 4; i++) {  for (int j = 4; j > i; j--) {  System.out.print("#");  }  System.out.println();  }  } |

* 3000米长的绳子，每天减一半。问多少天这个绳子会小于5米？不考虑小数。

|  |
| --- |
| package com.hfm.baisc.java;  /\*\*  \* 练习:3000米长的绳子，每天减一半。问多少天这个绳子会小于5米？不考虑小数。  \*  \* @author hfm  \* @version 1.01 2020-03-19 11:18  \* @date 2020/3/19  \*/  public class GetDay {  public static void main(String[] args) {  int n = 3000;  for (int i = 0; ; i++) {  n /= 2;  if (n < 5) {  System.out.println(i);  break;  }  }  }  } |

* 打印1-100之间非13的倍数，使用continue语句

|  |
| --- |
| private static void print13Dis() {  for (int i = 1; i <=100 ; i++) {  if(i%13 ==0){  continue;  }else {  System.out.println(i);  }  }  } |

* 用循环控制语句打印输出：1+3+5+...+99=?的结果

|  |
| --- |
| public static void main(String[] args) {  int sum = 0;  for (int i = 1; i <= 99; i+=2) {  sum += i;  }  System.out.println("sum = " + sum);  } |
| public static void main(String[] args) {  int sum = 0;  for (int i = 1; i < 100; i++) {  if (i % 2 != 0) {  sum += i;  }  }  System.out.println("sum = " + sum);  } |

* 写出结果

|  |
| --- |
| public class Demo{  public static void main(String []args){  int i = 0, j = 5;  tp: for (;;){  i++;  for(;;){  if(i > j--)  break tp;  }  }  // i=1,j=-1;  System.out.println("i = " + i + ", j = "+ j);  }  } |

* 输出从1到100之间所有不能被3整除的数;并输出这些整数的和

|  |
| --- |
| private static void get3Sum() {  int sum = 0;  for (int i = 1; i <= 100; i++) {  if (i % 3 != 0) {  sum += i;  } else {  continue;  }  }  System.out.println(sum);  } |

* 猜数字游戏

随机生成一个100以内的数，猜数字游戏：从键盘输入数，如果大了提示，大了，如果小了，提示小了，如果对了，就不再猜了，并统计一共猜了多少次？

提示：随机数

|  |
| --- |
| import java.util.Random;  Random rand = new Random();  int num= rand.nextInt(100);  import java.util.Random;  import java.util.Scanner;  public class TestDoWhileLoop{  public static void main(String[] args){  //1、随机产生一个100以内的整数  Random rand = new Random();  //int num = rand.nextInt();//产生的是任意大小的整数  int num = rand.nextInt(100);//产生[0,100)的整数  System.out.println(num);    //2、键盘输入  Scanner input = new Scanner(System.in);    //声明变量  int guess;  int count = 0;  do{  //循环体至少执行一次  System.out.print("请输入一个整数：");  guess = input.nextInt();//为变量赋值    count++;//输入一次，计数一次    if(guess>num){  System.out.println("大了");  }else if(guess < num){  System.out.println("小了");  }else{  System.out.println("猜对了");  }  }while(guess != num);    System.out.println("一共猜了：" + count + "次");  }  } |

* 打印1-100之间数，其中3、5、7的倍数不打印（continue）

|  |
| --- |
| class TestContinue {  public static void main(String[] args) {  //打印1-100之间数，其中3、5、7的倍数不打印  for(int i=1; i<=100; i++){  if(i%3==0 | i%5==0 | i%7==0){  //跳过下面的打印语句，提前进入下一次循环，即i++语句  //break;//结束循环  continue;  }    //下面这部分循环体语句，有些情况下需要跳过  System.out.println(i);  }  System.out.println("over");  }  } |

* 输入两个正整数m和n，求其最大公约数和最小公倍数

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:输入两个正整数m和n，求其最大公约数和最小公倍数  \*  \* @author hfm  \* @version 1.01 2020-03-19 11:36  \* @date 2020/3/19  \*/  public class MultiplesAndDivisors {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入正整数a");  int a = scanner.nextInt();  System.out.println("正整数a 的值为" + a);  System.out.println("请输入正整数b");  int b = scanner.nextInt();  System.out.println("正整数a 的值为" + b);  int max = (a > b) ? a : b;  int min = (a < b) ? a : b;  int minyue = 0;  for (int i = min - 1; i >= 1; i--) {  if (a % i == 0 && b % i == 0) {  System.out.println("最大公约数为" + i);  minyue = i;  break;  }  }  for (int i = max; ; i++) {  if (i % a == 0 && i % b == 0) {  System.out.println("最小公倍数为" + i);  break;  }  }    int num = max / minyue;  for (int i = num; ; i++) {  if (i \* minyue % a == 0 && i % b == 0) {  System.out.println("最小公倍数为" + i);  break;  }  }  }  } |

* 打印自然数的个数

1）打印1~100之间 6的倍数的个数

2）求出1～100之间，既是3又是7的倍数的自然数出现的次数？

|  |
| --- |
| public static void main(String[] args) {  int count1 = 0,count2 = 0;  for (int x = 1; x <= 100; x++) {  if (x % 6 == 0){  count1++;  }  if(x % 3 == 0 && x % 7 == 0){  count2++;  }    }  System.out.println("count1=" + count1);  System.out.println("count2=" + count2);  } |

* 求调和级数中从第多少项开始值大于10

调和级数的第n项形式为：1+1/2+1/3+…+1/n

|  |
| --- |
| public static void main(String[] args) {  double sum = 0.0;  int i = 1;  while (true) {  sum += 1.0 / i;  if (sum > 10) {  break;  }  i++;  }  System.out.println(i);  } |

* 打印如下的图形：三角形

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

|  |
| --- |
| private static void print5() {  for (int i = 0; i < 7; i++) {  if (i < 4) {  for (int j = 0; j < 2 \* i + 1; j++) {  System.out.print("\* ");  }  System.out.println();  } else {  for (int k = 0; k < 13 - 2 \* i; k++) {  System.out.print("\* ");  }  System.out.println();  }  }  } |

* 打印如下的图形

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

|  |
| --- |
| private static void print6() {  for (int i = 0; i < 5; i++) {  // 输出“-”  for (int j = 0; j < 4 - i; j++) {  System.out.print(" ");  }  // 输出“\* ”  for (int k = 0; k < i + 1; k++) {  System.out.print("\* ");  }  System.out.println();  }  // 下半部分  for (int i = 0; i < 4; i++) {  for (int j = 0; j < i + 1; j++) {  System.out.print(" ");  }  for (int k = 0; k < 4 - i; k++) {  System.out.print("\* ");  }  System.out.println();  }  } |

* 打印如下的图形

\*\*\*\*\*\*\*\*\*\*

\*\*\*\* \*\*\*\*

\*\*\* \*\*\*

\*\* \*\*

\* \*

\*\* \*\*

\*\*\* \*\*\*

\*\*\*\* \*\*\*\*

\*\*\*\*\*\*\*\*\*\*

|  |
| --- |
| private static void print7() {  System.out.println("\*\*\*\*\*\*\*\*\*\* \n" +  "\*\*\*\* \*\*\*\* \n" +  "\*\*\* \*\*\* \n" +  "\*\* \*\* \n" +  "\* \* \n" +  "\*\* \*\*\n" +  "\*\*\* \*\*\*\n" +  "\*\*\*\* \*\*\*\*\n" +  "\*\*\*\*\*\*\*\*\*\*\n");  } |

* 打印如下的图形：心形

\*\*\*\*\*\* \*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*

\*

|  |
| --- |
| class Heart {  public static void main(String[] args) {  // 分三个大部分 上中下  for (int i = 0, k = 0; i < 14; i++) {// 打印行  // 上部分 上分为 四个部分  if (i < 3) {  for (int j = 0; j < 5 - 2 \* i; j++) {// 1、空心  System.out.print(" ");  }  if (i == 2) {// 2、\*  for (int j = 0; j < 6 + 4 \* i - 1; j++) {  System.out.print("\*");  }  for (int j = 0; j < 7 - 4 \* i + 2; j++) {// 3、空心  System.out.print(" ");  }  for (int j = 0; j < 6 + 4 \* i - 1; j++) {// 4、\*  System.out.print("\*");  }  } else {  for (int j = 0; j < 6 + 4 \* i; j++) {// 2、\*  System.out.print("\*");  }  for (int j = 0; j < 7 - 4 \* i; j++) {// 3、空心  System.out.print(" ");  }  for (int j = 0; j < 6 + 4 \* i; j++) {// 4、\*  System.out.print("\*");  }  }  } else if (i < 6) {// 中间  for (int j = 0; j < 29; j++) {  System.out.print("\*");  }  } else {// 下部分 6  if (i == 13) {  for (int j = 0; j < 2 \* (i - 6); j++) {// 打印空格  System.out.print(" ");  }  System.out.print("\*");  } else {  for (int j = 0; j < 2 \* (i - 6) + 1; j++) {// 打印空格  System.out.print(" ");  }  for (int j = 1; j < 28 - 4 \* k; j++) {  System.out.print("\*");  }  k++;  }  }  System.out.println();// 换行  }  }  } |

* 编写程序，打印100-200之间的质数

|  |
| --- |
| package com.hfm.baisc.java;  /\*\*  \* 练习: 编写程序，打印100-200之间的质数  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class PrimeNumber {  public static void main(String[] args) {  Outer:  for (int i1 = 2; i1 <= 100; i1++) {  Inner:  for (int j1 = 2; j1 <= Math.sqrt(i1); j1++) {  if (i1 % j1 == 0) {  continue Outer;  }  }  System.out.println(i1);  }  }  } |

* 一个数如果恰好等于它的因子之和，这个数就称为"完数"（因子：除去这个数本身正的约数）

例如6=1＋2＋3.编程 找出1000以内的所有完数

|  |
| --- |
| public class WanShu {  static int count;  public static void main(String[] args) {  for (int i = 1; i <= 1000; i++) {  int factor = 0;  for (int j = 1; j < i; j++) {  if (i % j == 0)  factor += j;  }  if (factor == i) {  System.out.println(i);  count++;  }  }  System.out.println("1-1000之间的完数个数为：" + count);  }  } |

* 写一个程序，找出4位数的所有吸血鬼的数字

例如：1260=21\*60

1827=21\*87

|  |
| --- |
| public class Test1 {  public static void main(String[] args) {  for (int num = 1001; num < 10000; num++) {  math(num);  }  }  public static void math(int num) {  int[] temp1 = new int[2];  int[] temp2 = new int[2];  int a = num / 1000;  int b = num / 100 % 10;  int c = num / 10 % 10;  int d = num % 10;  int[] data = { a, b, c, d };  for (int i = 0; i < data.length; i++) {  for (int j = 0; j < data.length; j++) {  if (i == j) {  continue;  }  temp1[0] = data[i];  temp1[1] = data[j];  for (int m = 0; m < data.length; m++) {  if (m != i && m != j) {  temp2[0] = data[m];  for (int n = 0; n < data.length; n++) {  if (n != i && n != j && n != m) {  temp2[1] = data[n];  multi(data, temp1, temp2);  }  }  }  }  }  }  }  public static int toInt(int[] temp) {  int m = 0;  int[] temp1 = new int[temp.length];  for (int i = 0; i < temp.length; i++) {  temp1[i] = temp[i] \* (int) Math.pow(10, temp.length - 1 - i);  }  for (int i = 0; i < temp1.length; i++) {  m += temp1[i];  }  return m;  }  public static void multi(int[] temp, int[] temp1, int[] temp2) {  int i = toInt(temp1);  int j = toInt(temp2);  int k = toInt(temp);  if (k == i \* j) {  System.out.println(k + "=" + i + "\*" + j);  }  }  } |

* 山上有一口缸可以装50升水，现在有15升水。老和尚叫小和尚下山挑水，每次可以挑5升。问：小和尚要挑几次水才可以把水缸挑满？通过编程解决这个问题。

提示：

(1) 用整型变量water表示水缸里的水“int water = 15;”。

(2) 用整型变量l表示小和尚下山挑水的次数“int l = 0;”。

(3) 分析循环条件（水少于50升），循环操作（水增加5升，挑水次数增加1）。

(4) 套用while循环（或do-while循环）写出代码。

|  |
| --- |
| package com.hfm.baisc.java;  /\*\*  \* 练习:山上有一口缸可以装50升水，现在有15升水。老和尚叫小和尚下山挑水，每次可以挑5升。问：小和尚要挑几次水才可以把水缸挑满？通过编程解决这个问题。  \* 提示：  \* (1) 用整型变量water表示水缸里的水“int water = 15;”。  \* (2) 用整型变量l表示小和尚下山挑水的次数“int l = 0;”。  \* (3) 分析循环条件（水少于50升），循环操作（水增加5升，挑水次数增加1）。  \* (4) 套用while循环（或do-while循环）写出代码。  \*  \* @author hfm  \* @version 1.01 2020-03-19 12:31  \* @date 2020/3/19  \*/  public class CarryWater {  public static void main(String[] args) {  int num = 0;  int all = 0;  for (int i = 0; ; i++) {  if (15 + i \* 5 >= 50) {  System.out.println(i);  break;  }  }  }  } |

* 生成13位条形码, Ean-13码规则：第十三位数字是前十二位数字经过计算得到的校验码。

例如：690123456789

计算其校验码的过程为：

1. 前十二位的奇数位和6+0+2+4+6+8=26
   1. ② 前十二位的偶数位和9+1+3+5+7+9=34
2. 将奇数和与偶数和的三倍相加26+34\*3=128
   1. ④ 取结果的个位数：128的个位数为8

⑤ 用10减去这个个位数10-8=2所以校验码为2

（注：如果取结果的个位数为0，那么校验码不是为10（10-0=10），而是0）

实现上述代码的功能，计算验证码，输入12位条码，返回带验证码的条码。

例：输入：692223361219输出：6922233612192

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习: 生成13位条形码, Ean-13码规则：第十三位数字是前十二位数字经过计算得到的校验码。  \* 例如：690123456789  \* 计算其校验码的过程为：  \* ① 前十二位的奇数位和6+0+2+4+6+8=26  \* a) ② 前十二位的偶数位和9+1+3+5+7+9=34  \* ② 将奇数和与偶数和的三倍相加26+34\*3=128  \* a) ④ 取结果的个位数：128的个位数为8  \* ⑤ 用10减去这个个位数10-8=2所以校验码为2  \* （注：如果取结果的个位数为0，那么校验码不是为10（10-0=10），而是0）  \* 实现上述代码的功能，计算验证码，输入12位条码，返回带验证码的条码。  \* 例：输入：692223361219输出：6922233612192  \*  \* @author hfm  \* @version 1.01 2020-03-19 12:58  \* @date 2020/3/19  \*/  public class VerificationCode {  public static void main(String[] args) {  System.out.println("请输入12位长整数");  Scanner scanner = new Scanner(System.in);  Long num = scanner.nextLong();  Long temp = num;  System.out.println("输入的长整数为" + num);  int oddSum = 0;  temp /= 10;  oddSum += temp % 10;  while (temp > 0) {  temp /= 100;  oddSum += (temp % 10);  }  System.out.println(oddSum);  temp = num;  int evenSum = 0;  evenSum += temp % 10;  while (temp > 0) {  temp /= 100;  evenSum += (temp % 10);  }  System.out.println(evenSum);  int num13 = 10 - (oddSum \* 3 + evenSum) % 10;  if(num13 ==10){  num13 = 0;  }  System.out.println(num + Integer.valueOf(num13).toString());  }  } |

* 实现判断一个4位整数，统计出此整数里面包含多少个偶数，多少个奇数的功能

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:实现判断一个4位整数，统计出此整数里面包含多少个偶数，多少个奇数的功能  \*  \* @author hfm  \* @version 1.01 2020-03-19 13:36  \* @date 2020/3/19  \*/  public class JudgeNum {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入一个4位正整数");  int num = scanner.nextInt();  int count = 0;  int temp = num;  while (temp > 0) {  if ((temp % 10) % 2 == 0) {  count++;  }  temp /= 10;  }  System.out.println("正整数" + num + "中有" + count + "个偶数 有" + (4 - count) + "个奇数");  }  } |

* 开发一款软件，根据公式（身高-108）\*2=体重，可以有10斤左右的浮动。来观察测试者体重是否合适。

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:开发一款软件，根据公式（身高-108）\*2=体重，可以有10斤左右的浮动。来观察测试者体重是否合适。  \*  \* @author hfm  \* @version 1.01 2020-03-19 13:48  \* @date 2020/3/19  \*/  public class JudgeFat {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入体重");  int weigth = scanner.nextInt();  System.out.println("输入的体重为" + weigth);  System.out.println("请输入身高");  int heigth = scanner.nextInt();  System.out.println("输入的身高为" + heigth);  if (Math.abs((heigth - 108) \* 2 - weigth) < 10) {  System.out.println("测试者体重合适");  }else{  System.out.println("测试者体重不合适");  }  }  } |

* 有3个整数，给出提示信息：

能否创建三角形；

两边之和大于第三边 三个条件都要写

如果能构建三角形，提示是直角三角形还是等边三角形等腰三角形还是普通三角形；最后输出三角形面积；

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:有3个整数，给出提示信息：  \* 能否创建三角形；  \* 两边之和大于第三边 三个条件都要写  \* 如果能构建三角形，提示是直角三角形还是等边三角形等腰三角形还是普通三角形；最后输出三角形面积；  \*  \* @author hfm  \* @version 1.01 2020-03-19 14:03  \* @date 2020/3/19  \*/  public class JudgeTriangle {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请依次输入三个数并用, 号隔开");  String arr = scanner.next();  String[] arrs = arr.split(",");  int a = Integer.parseInt(arrs[0]);  int b = Integer.parseInt(arrs[1]);  int c = Integer.parseInt(arrs[2]);  if (a + b > c || a + c > b || b + c > a) {  System.out.println(arr + "三个数可以创建三角形");  } else {  System.out.println(arr + "三个数不可以创建三角形");  }  if ((a + b > c || a + c > b || b + c > a) && (Math.pow(a, 2) + Math.pow(b, 2) == Math.pow(c, 2) || Math.pow(b, 2) + Math.pow(c, 2) == Math.pow(a, 2) || Math.pow(a, 2) + Math.pow(c, 2) == Math.pow(b, 2))) {  System.out.println("可以创建直角三角形");  }  if ((a + b > c || a + c > b || b + c > a) && a == b && b == c) {  System.out.println("可以创建等腰三角形");  }  System.out.println("三角形的面积为" + Math.sqrt((a + b + c) \* (a + b - c) \* (a + c - b) \* (b + c - a)) / 4);  }  } |

* 在JAVA中，如何跳出当前的多重嵌套循环？

答：用break或return 的方式。

* “&”和“&&”的异同

相同: 运算结果都是相同的

不同: && 的效率更高, 当左边不满足要求时, 立即得出结果

* 程序输出

|  |
| --- |
| class OperatorTest {  public static void main(String[] args) {  boolean x = true;  boolean y = false;  short z = 40;  if ((z++ == 40) && (y = true)) {  z++;  }  if ((x = false) || (++z == 43)) {  z++;  }  System.out.println("z = " + z);  }  } |

结果为: 46

* 定义三个int型变量并赋值，使用三元运算符或者if-else获取这三个数中的较大数的实现

|  |
| --- |
| int num1 = 10,num2 = 21,num3 = -21;  int max;  if(num1 >= num2 && num1 >= num3){  max = num1;  }else if(num2 >= num1 && num2 >= num3){  max = num2;  }else{  max = num3;  }  System.out.println("三个数中的最大值为：" + max); |

* 编写程序，声明2个double型变量并赋值。判断第一个数大于10.0，且第2个数小于20.0，打印两数之和。否则，打印两数的乘积。

|  |
| --- |
| double d1 = 12.3;  double d2 = 32.1;  if(d1 > 10 && d2 < 20){  }else{    } |

* switch后面使用的表达式可以是哪些数据类型的。

byte 、 short 、char 、int 、枚举类型变量、String类型。

* 使用switch语句改写下列if语句

|  |
| --- |
| int a = 3;  int x = 100;  if(a==1)  x+=5;  else if(a==2)  x+=10;  else if(a==3)  x+=16;  else  x+=34; |
| int a = 3;  int x = 100;  switch(a){  case 1:  x+=5;  break;  case 2:  x+=10;  break;  case 3:  x+=16;  break;  default:  x+=34;  //break;  } |

* 谈谈你对三元运算符、if-else和switch-case结构使用场景的理解

三元运算符与可以转化为if-else语句, 但是if-else语句不一定可以转化为三元运算符.

对于分支结构较少的可以使用switch语句, 并且运行效率更高,

* 如何从控制台获取String和int型的变量，并输出？使用代码实现

|  |
| --- |
| 1.import java.util.Scanner;  2. Scanner scan = new Scanner(System.in);  3. String info = scan.next();  int id = scan.nextInt(); |

* 岳小鹏参加Java考试，他和父亲岳不群达成承诺：如果成绩为100分时，奖励一辆BMW；成绩为(80，99]时，奖励一台iphone xs max；当成绩为[60,80]时，奖励一个 iPad；其它时，什么奖励也没有。请从键盘输入岳小鹏的期末成绩，并加以判断

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习 .给定一个百分制的分数，输出相应的等级。  \* 90分以上 A级  \* 80~89 B级  \* 70~79 C级  \* 60~69 D级  \* 60分以下 E级  \*  \* 练习:岳小鹏参加Java考试，他和父亲岳不群达成承诺：  \* 如果：  \* 成绩为100分时，奖励一辆BMW；  \* 成绩为(80，99]时，奖励一台iphone xs max；  \* 当成绩为[60,80]时，奖励一个 iPad；  \* 其它时，什么奖励也没有。  \* 请从键盘输入岳小鹏的期末成绩，并加以判断  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class GetGrade {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入成绩");  double grade = scanner.nextDouble();  if (grade < 60 && grade >= 0) {  System.out.println("成绩为E");  }  if (grade >= 60 && grade < 70) {  System.out.println("成绩为D");  }  if (grade >= 70 && grade < 80) {  System.out.println("成绩为C");  }  if (grade >= 80 && grade < 90) {  System.out.println("成绩为B");  }  if (grade >= 90 && grade <= 100) {  System.out.println("成绩为A");  }  if (grade < 0 || grade > 100) {  System.out.println("输入错误");  }  if(grade == 100){  System.out.println("奖励一辆BMW");  }  if(grade>80 && grade<=99){  System.out.println("奖励一台iphone xs max");  }  if(grade>60 && grade<=80){  System.out.println("奖励一个 iPad");  }  if(grade<=60){  System.out.println("nothing");  }  }  } |

* 使用for循环遍历100以内的奇数，并计算所有的奇数的和并输出。

|  |
| --- |
| int sum = 0;  for(int i = 1;i <= 100;i++){  if(i % 2 != 0){  Sysout(i);  sum += i;  }  }  syout(sum); |

* 循环结构是如何最后退出循环的，有哪些不同的情况请说明。

1. 循环条件返回false

② 在循环体内，一旦执行到break，跳出循环

* 指出如下程序输出的结果

|  |
| --- |
| label: for (int i = 1; i <= 4; i++) {  for (int j = 1; j <= 10; j++) {  if (j % 4 == 0) {  continue label;  }  System.out.print(j);  }  System.out.println();  } |

正确结果：123123123123

* 对下列代码，若有输出，指出输出结果。

|  |
| --- |
| int x = 4;  int y = 1;  if (x > 2) {  if (y > 2)  System.out.println(x + y);  System.out.println("atguigu");  } else  System.out.println("x is " + x); |
| boolean b = true;  //如果写成if(b=false)能编译通过吗？如果能，结果是？  if(b == false)  System.out.println("a");  else if(b)  System.out.println("b");  else if(!b)  System.out.println("c");  else  System.out.println("d"); |

* 编写程序，声明2个int型变量并赋值。判断两数之和，如果大于等于50，打印“hello world!”

|  |
| --- |
| Public static void printHello{  Scanner scanner = new Scanner(System.in);  System.out.println(“请输入正整数a”);  Int a = scanner.nextInt();  System.out.println(“请输入正整数b”);  Int b = scanner.nextInt();  If ((a + b) > 50){  System.out.println(“Hello World!”);  }  } |

* 编写程序，声明2个double型变量并赋值。判断第一个数大于10.0，且第2个数小于20.0，打印两数之和。否则，打印两数的乘积。

|  |
| --- |
| Public static void main(String [] args){  Scanner scanner = new Scanner(System.in);  System.out.println(“请输入数a”);  Double a = scanner.next Double ();  System.out.println(“请输入数b”);  Double b = scanner.next Double ();  If(a >10.0 && b<20.0){  System.out.println(a+b);  }else{  System.out.println(a\*b);  }  } |

* 我家的狗5岁了，5岁的狗相当于人类多大呢？其实，狗的前两年每一年相当于人类的10.5岁，之后每增加一年就增加四岁。那么5岁的狗相当于人类多少年龄呢？应该是：10.5 + 10.5 + 4 + 4 + 4 = 33岁。编写一个程序，获取用户输入的狗的年龄，通过程序显示其相当于人类的年龄。如果用户输入负数，请显示一个提示信息。

|  |
| --- |
| Public static void getAge(int age){  Int result =0;  If(age<=2){  System.out.println(“狗的年龄为”+(age \* 10.5));  }else{  System.out.println(“狗的年龄为”+(2 \* 10.5 + (age -2)\*5));  }  } |

* 大家都知道，男大当婚，女大当嫁。那么女方家长要嫁女儿，当然要提出一定的条件：高：180cm以上；富：财富1千万以上；帅：是。如果这三个条件同时满足，则：“我一定要嫁给他!!!”如果三个条件有为真的情况，则：“嫁吧，比上不足，比下有余。”如果三个条件都不满足，则：“不嫁！”

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:大家都知道，男大当婚，女大当嫁。那么女方家长要嫁女儿，当然要提出一定的条件：高：180cm以上；富：财富1千万以上；帅：是。如果这三个条件同时满足，则：“我一定要嫁给他!!!”如果三个条件有为真的情况，则：“嫁吧，比上不足，比下有余。”如果三个条件都不满足，则：“不嫁！  \*  \* @author hfm  \* @version 1.01 2020-03-19 15:08  \* @date 2020/3/19  \*/  public class JudgeMarry {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("输入身高");  double heigth = scanner.nextDouble();  System.out.println("输入财富值");  double money = scanner.nextDouble();  System.out.println("输入帅气值");  double shuai = scanner.nextDouble();  if (heigth > 180 && money > 1000\_0000.0 && shuai > 60.0) {  System.out.println("我一定要嫁给他!!!");  } else if (heigth > 180 || money > 1000\_0000.0 || shuai > 60.0) {  System.out.println("嫁吧，比上不足，比下有余。");  } else {  System.out.println("不嫁");  }  }  } |

* 使用 switch 把小写类型的 char型转为大写。只转换 a, b, c, d, e. 其它的输出 “other”。

提示：String word = scan.next(); char c = word.charAt(0); switch(c){}

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习: 使用 switch 把小写类型的 char型转为大写。只转换 a, b, c, d, e. 其它的输出 “other”。  \* 提示：String word = scan.next(); char c = word.charAt(0); switch(c){}  \*  \* @author hfm  \* @version 1.01 2020-03-19 15:15  \* @date 2020/3/19  \*/  public class SwitchTest {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("输入一个字符");  String aChar = scanner.next();  switch (aChar) {  case "a":  case "b":  case "c":  case "d":  case "e":  System.out.println(aChar.charAt(0));  break;  default:  System.out.println("other");  }  }  } |

* 编写程序： 从键盘分别输入年、月、日，判断这一天是当年的第几天

注：判断一年是否是闰年的标准：1）可以被4整除，但不可被100整除或2）可以被400整除

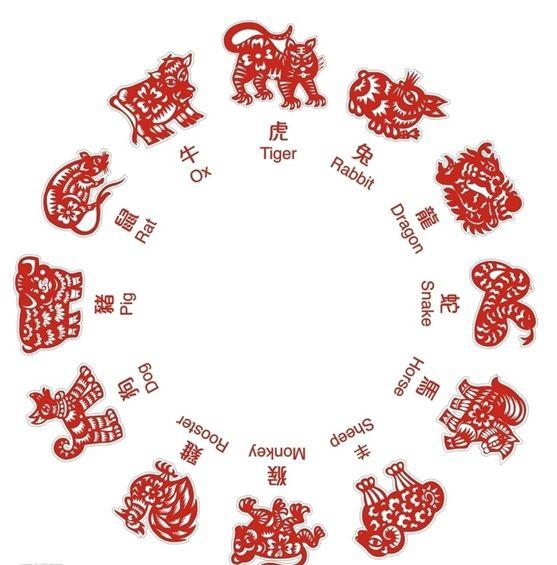
|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Scanner;  /\*\*  \* 练习:从键盘分别输入年、月、日，判断这一天是当年的第几天  \* 注：判断一年是否是闰年的标准：1）可以被4整除，但不可被100整除或2）可以被400整除  \*  \* @author hfm  \* @version 1.01 2020-03-19 15:28  \* @date 2020/3/19  \*/  public class JudgeDay {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入年/月/日并用/号隔开");  String time = scanner.next();  String[] times = time.split("/");  int year = Integer.parseInt(times[0]);  int month = Integer.parseInt(times[1]);  int day = Integer.parseInt(times[2]);  if (year % 4 == 0 && year % 100 != 0 || year % 400 == 0) {  getDay(month, day, 29);  } else {  getDay(month, day, 28);  }  }  private static void getDay(int month, int day, int i) {  if (month == 1) {  System.out.println("这一天是当年的第" + day + "天");  }  if (month == 2) {  System.out.println("这一天是当年的第" + (day + 31) + "天");  }  if (month == 3) {  System.out.println("这一天是当年的第" + (day + 31 + i) + "天");  }  if (month == 4) {  System.out.println("这一天是当年的第" + (day + 31 \* 2 + i) + "天");  }  if (month == 5) {  System.out.println("这一天是当年的第" + (day + 31 \* 2 + i + 30) + "天");  }  if (month == 6) {  System.out.println("这一天是当年的第" + (day + 31 \* 3 + i + 30) + "天");  }  if (month == 7) {  System.out.println("这一天是当年的第" + (day + 31 \* 3 + i + 30 \* 2) + "天");  }  if (month == 8) {  System.out.println("这一天是当年的第" + (day + 31 \* 4 + i + 30 \* 2) + "天");  }  if (month == 9) {  System.out.println("这一天是当年的第" + (day + 31 \* 5 + i + 30 \* 2) + "天");  }  if (month == 10) {  System.out.println("这一天是当年的第" + (day + 31 \* 5 + i + 30 \* 3) + "天");  }  if (month == 11) {  System.out.println("这一天是当年的第" + (day + 31 \* 5 + i + 30 \* 4) + "天");  }  if (month == 12) {  System.out.println("这一天是当年的第" + (day + 31 \* 5 + i + 30 \* 5) + "天");  }  }  } |

* 使用switch语句改写下列if语句

|  |
| --- |
| int a = 3;  int x = 100;  if(a==1)  x+=5;  else if(a==2)  x+=10;  else if(a==3)  x+=16;  else  x+=34; |
| int a = 3;  int x = 100;  Switch(a){  Case 1: x+=5;break;  Case 2: x+=10;break;  Case 3: x+=16;break;  Default: x+=34;  } |

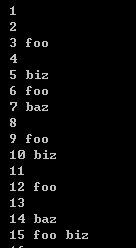
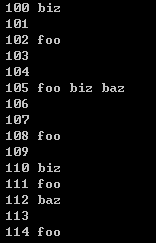
* 编写一个程序，为一个给定的年份找出其对应的中国生肖。中国的生肖基于12年一个周期，

每年用一个动物代表：rat、ox、tiger、rabbit、dragon、snake、horse、sheep、monkey、rooster、dog、pig。提示：2019 年：猪 2019 % 12 == 3



|  |
| --- |
| Switch(year %12){  Case 0:System.out.println(“rat”);break;  Case 1:System.out.println(“ox”);break;  Case 2:System.out.println(“tiger”);break;  Case 3:System.out.println(“rabbit”);break;  Case 4:System.out.println(“dragon”);break;  Case 5:System.out.println(“snake”);break;  Case 6:System.out.println(“horse”);break;  Case 7:System.out.println(“sheep”);break;  Case 8:System.out.println(“monkey”);break;  Case 9:System.out.println(“rooster”);break;  Case 10:System.out.println(“dog”);break;  Case 11:System.out.println(“pig”);break;  Default: System.out.println();  } |

* 编写程序从1循环到150，并在每行打印一个值，另外在每个3的倍数行上打印出“foo”,在每个5的倍数行上打印“biz”,在每个7的倍数行上打印输出“baz”。

|  |
| --- |
| For(int I = 0;i<150;i++){  If(i%3 == 0){  System.out.println(“foo”);  }  If(i%5==0){  System.out.println(“biz”);  }  If(i%7==0){  System.out.println(“baz”);  }  } |

* 说明：break关键字的使用

Break 用在switch 语句与循环语句中, 表示退出循环或退出switch语句

* 打印1~100之间所有是7的倍数的整数的个数及总和（体会设置计数器的思想）

|  |
| --- |
| package com.hfm.baisc.java;  /\*\*  \* 练习 编写程序输出1-100之间所有能被7整除的偶数。  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class Get7Divisible {  public static void main(String[] args) {  for (int i = 0; i <= 99; i++) {  if (i % 7 == 0) {  System.out.println(i + "能够被7整除！");  }  }  }  } |

* 输出所有的水仙花数，所谓水仙花数是指一个3位数，其各个位上数字立方和等于其本身。例如： 153 = 1\*1\*1 + 3\*3\*3 + 5\*5\*5

|  |
| --- |
| package com.hfm.baisc.java;  /\*\*  \* 练习:输出所有的水仙花数，所谓水仙花数是指一个3位数，其各个位上数字立方和等于其本身。例如： 153 = 1\*1\*1 + 3\*3\*3 + 5\*5\*5  \*  \* @author hfm  \* @version 1.01 2020-03-19 15:59  \* @date 2020/3/19  \*/  public class FlowerNum {  public static void main(String[] args) {  for (int i = 100; i < 999; i++) {  if (i == (Math.pow(i % 10, 3) + Math.pow(i / 10 % 10, 3) + Math.pow(i / 100, 3))) {  System.out.println(i);  }  }  }  } |

* 从键盘读入个数不确定的整数，并判断读入的正数和负数的个数，输入为0时结束程序。最简单“ 无限” 循环：格式：while(true) , for(;;), 无限循环存在的原因是并不知道循环多少次，需要根据循环体内部某些条件，来控制循环的结束。

|  |
| --- |
| class PositiveNegative {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  int positiveNumber = 0;//统计正数的个数  int negativeNumber = 0;//统计负数的个数  for(;;){ //while(true){  System.out.println("请输入一个整数：");  int z = scanner.nextInt();  if(z>0)  positiveNumber++;  else if(z<0)  negativeNumber++;  else  break;  }  System.out.println("正数的个数为："+ positiveNumber);  System.out.println("负数的个数为："+ negativeNumber);  }  } |

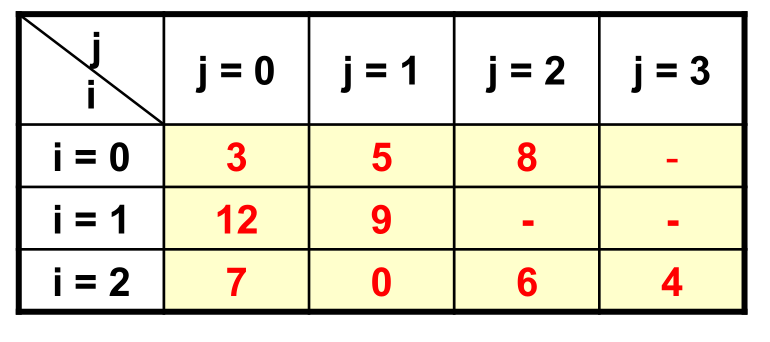
* 100以内的所有质数的输出。质数：素数，只能被1和它本身整除的自然数。-->从2开始，到这个数-1结束为止，都不能被这个数本身整除。最小的质数是：2

|  |
| --- |
| \*/  class PrimeNumberTest {  public static void main(String[] args) {  boolean isFlag = true;//标识i是否被j除尽，一旦除尽，修改其值  for(int i = 2;i <= 100;i++){//遍历100以内的自然数  for(int j = 2;j < i;j++){//j:被i去除  if(i % j == 0){ //i被j除尽  isFlag = false;  }  }  //  if(isFlag == true){  System.out.println(i);  }  //重置isFlag  isFlag = true;  }  }  } |
| /\*  100000以内的所有质数的输出。实现方式一  质数：素数，只能被1和它本身整除的自然数。-->从2开始，到这个数-1结束为止，都不能被这个数本身整除。  对PrimeNumberTest.java文件中质数输出问题的优化  \*/  class PrimeNumberTest1 {  public static void main(String[] args) {  boolean isFlag = true;//标识i是否被j除尽，一旦除尽，修改其值  int count = 0;//记录质数的个数  //获取当前时间距离1970-01-01 00:00:00 的毫秒数  long start = System.currentTimeMillis();  for(int i = 2;i <= 100000;i++){//遍历100000以内的自然数  //优化二：对本身是质数的自然数是有效的。  //for(int j = 2;j < i;j++){  for(int j = 2;j <= Math.sqrt(i);j++){//j:被i去除  if(i % j == 0){ //i被j除尽  isFlag = false;  break;//优化一：只对本身非质数的自然数是有效的。  }  }  //  if(isFlag == true){  //System.out.println(i);  count++;  }  //重置isFlag  isFlag = true;  }  //获取当前时间距离1970-01-01 00:00:00 的毫秒数  long end = System.currentTimeMillis();  System.out.println("质数的个数为：" + count);  System.out.println("所花费的时间为：" + (end - start));//17110 - 优化一：break:1546 - 优化二：13  }  } |
| /\*  100000以内的所有质数的输出。实现方式二  质数：素数，只能被1和它本身整除的自然数。-->从2开始，到这个数-1结束为止，都不能被这个数本身整除。  对PrimeNumberTest.java文件中质数输出问题的优化  \*/  class PrimeNumberTest2 {  public static void main(String[] args) {  int count = 0;//记录质数的个数  //获取当前时间距离1970-01-01 00:00:00 的毫秒数  long start = System.currentTimeMillis();  label:for(int i = 2;i <= 100000;i++){//遍历100000以内的自然数  for(int j = 2;j <= Math.sqrt(i);j++){//j:被i去除  if(i % j == 0){ //i被j除尽  continue label;  }  }  //能执行到此步骤的，都是质数  count++;  }  //获取当前时间距离1970-01-01 00:00:00 的毫秒数  long end = System.currentTimeMillis();  System.out.println("质数的个数为：" + count);  System.out.println("所花费的时间为：" + (end - start));//17110 - 优化一：break:1546 - 优化二：13  }  } |

* 升景坊单间短期出租4个月，550元/月（水电煤公摊，网费35元/月），空调、卫生间、厨房齐全。屋内均是IT行业人士，喜欢安静。所以要求来租者最好是同行或者刚毕业的年轻人，爱干净、安静。

|  |
| --- |
| public class ArrayTest {  public static void main(String[] args) {  int[] arr = new int[]{8,2,1,0,3};  int[] index = new int[]{2,0,3,2,4,0,1,3,2,3,3};  String tel = "";  for(int i = 0;i < index.length;i++){  tel += arr[index[i]];  }  System.out.println(" 联系方式：" + tel);  }  } |

* 获取arr数组中所有元素的和。提示：使用for的嵌套循环即可。



|  |
| --- |
| Int sum = 0;  For(int i=0;i<arr.length;i++){  For(int j=0;j<arr[i].length;i++){  Sum +=arr[i][j];  }  } |

* 声明：int[] x,y[]; 在给x,y变量赋值以后，以下选项允许通过编译的是：

|  |
| --- |
| a ) x[0] = y; no  b) y[0] = x; yes  c) y[0][0] = x; no  d) x[0][0] = y; no  e) y[0][0] = x[0]; yes  f) x = y; no |

提示：

一维数组：int[] x 或者int x[]

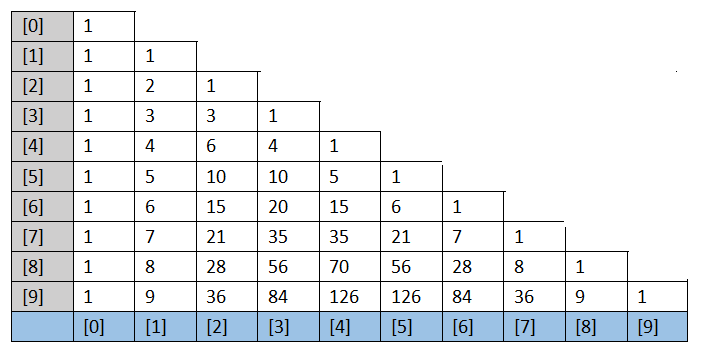
二维数组：int[][] y 者 或者 int[] y[] 或者 int y[][]

* 使用二维数组打印一个 10 行杨辉三角。

1. 第一行有 1 个元素, 第 n 行有 n 个元素

2. 每一行的第一个元素和最后一个元素都是 1

3. 从第三行开始, 对于非第一个元素和最后一个元素的元素。即：yanghui[i][j] = yanghui[i-1][j-1] + yanghui[i-1][j];



|  |
| --- |
| int[][] yanghui = new int[10][10];  for (int i = 0; i < yanghui.length; i++) {  for (int j = 0; j < i; j++) {  if (j == 0 || j == i - 1) {  yanghui[i][j] = 1;  } else {  yanghui[i][j] = yanghui[i - 1][j - 1] + yanghui[i - 1][j];  }  }  }  System.out.println(Arrays.deepToString(yanghui)); |

* 创建一个长度为6的int型数组，要求数组元素的值都在1-30之间，且是随机赋值。同时，要求元素的值各不相同。

|  |
| --- |
| class ArrayExer {  public static void main(String[] args) {  //方式一：  // int[] arr = new int[6];  // for (int i = 0; i < arr.length; i++) {// [0,1) [0,30) [1,31)  // arr[i] = (int) (Math.random() \* 30) + 1;  // boolean flag = false;  // while (true) {  // for (int j = 0; j < i; j++) {  // if (arr[i] == arr[j]) {  // flag = true;  // break;  // }  // }  // if (flag) {  // arr[i] = (int) (Math.random() \* 30) + 1;  // flag = false;  // continue;  // }  // break;  // }  // }  //  // for (int i = 0; i < arr.length; i++) {  // System.out.println(arr[i]);  // }  //方式二：  int[] arr = new int[6];  for (int i = 0; i < arr.length; i++) {// [0,1) [0,30) [1,31)  arr[i] = (int) (Math.random() \* 30) + 1;  for (int j = 0; j < i; j++) {  if (arr[i] == arr[j]) {  i--;  break;  }  }  }  for (int i = 0; i < arr.length; i++) {  System.out.println(arr[i]);  }  }  } |

* 定义一个int型的一维数组，包含10个元素，分别赋一些随机整数，然后求出所有元素的最大值，最小值，和值，平均值，并输出出来 。要求：所有随机数都是两位数。提示；[0,1) \* 90 [0,90) + 10 [10,100) [10,99] (int)(Math.random() \* 90 + 10)

|  |
| --- |
| Int [] arr = new int[10];  For(int i=0;i<arr.length;i++){  Arr[i] =(int) Math.random()\*90 +10;  }  Int sum =0;  Int max =0;  Int min =0;  For(int I =0;i<arr.length;i++){  If(arr[i]>max){  Max = arr[i];  }  If(arr[i]<min){  Min = arr[i];  }  Sum +=arr[i];  }  Int mindle = sum/arr.length; |

* 使用简单数组

(1)创建一个名为ArrayTest的类，在main()方法中声明array1和array2两个变量，他们是int[]类型的数组。

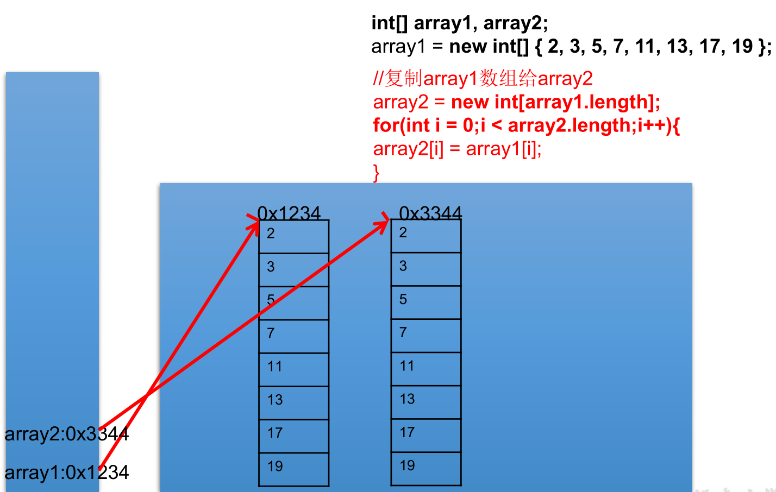
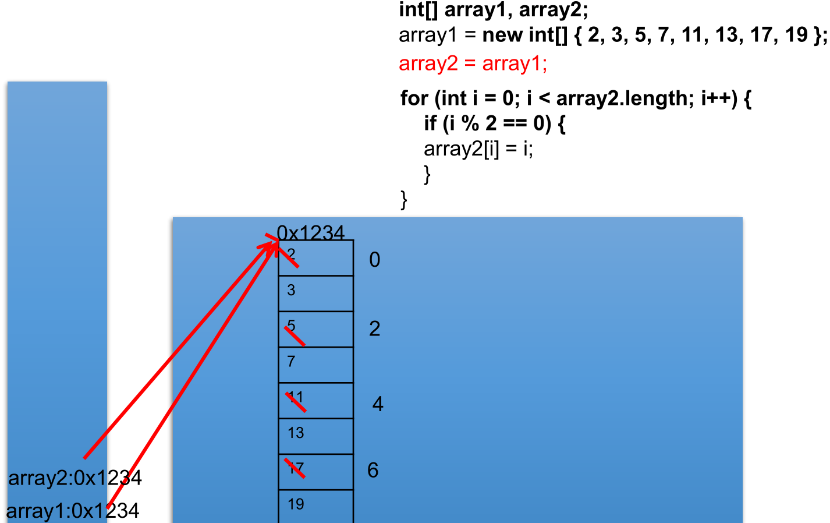
(2)使用大括号{}，把array1初始化为8个素数：2,3,5,7,11,13,17,19。

(3)显示array1的内容。

(4)赋值array2变量等于array1，修改array2中的偶索引元素，使其等于索引值(如array[0]=0,array[2]=2)。打印出array1。

思考：array1和array2是什么关系？

拓展：修改题目，实现array2对array1数组的复制



|  |
| --- |
| Public class ArrayTest(){  Public static void main(String [] args){  int [] array1 =new int[]{2,3,5,7,11,13,17,19};  int [] array2 =new int[]{};  System.out.println(Arrays.toString(array1));  array2 = array1;  array2[2] =2;  System.out.println(Arrays.toString(array1));  }  } |

* java能动态分配数组吗？

答：可以。int i = 12;int[] myInt = new int[i];

* 数组有没有length()这个方法? String有没有length()这个方法

答：数组没有length()这个方法，有length的属性。String有length()这个方法

* Java中的任何数据类型都可以使用System.out.pritln方法显示对基本数据类型而言，输出的往往是变量的值；对于像数组这一类复杂的数据类型，输出的是其堆空间中存储位置的hashCode值
* 操作二维数组的注意点: 操作二维数组不应使用常数来控制维数。具体方法是array.length表示行数，array[row].length来表示row行的列数。这样当数组行数和列数不相等时，代码可以自动调整为正确的值。
* 显示输出

|  |
| --- |
| // 各元素的值默认为null  String[] stringArray = new String[3];  // 对各元素进行初始化，但没有赋值。  for (int i = 0; i < stringArray.length; i++) {  stringArray[i] = new String();  System.out.println(stringArray[i]);  } |

答案：空 （有别于null）

* 回形数格式方阵的实现从键盘输入一个整数（1~20） 则以该数字为矩阵的大小，把1,2,3…n\*n 的数字按照顺时针螺旋的形式填入其中。例如： 输入数字2，则程序输出： 1 2 4 3 输入数字3，则程序输出： 1 2 3 8 9 4 7 6 5

输入数字4， 则程序输出：

1 2 3 4

12 13 14 5

11 16 15 6

10 9 8 7

|  |
| --- |
| 方式一：  class RectangleTest {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("输入一个数字");  int len = scanner.nextInt();  int[][] arr = new int[len][len];  int s = len \* len;  /\*  \* k = 1:向右 k = 2:向下 k = 3:向左 k = 4:向上  \*/  int k = 1;  int i = 0, j = 0;  for (int m = 1; m <= s; m++) {  if (k == 1) {  if (j < len && arr[i][j] == 0) {  arr[i][j++] = m;  } else {  k = 2;  i++;  j--;  m--;  }  } else if (k == 2) {  if (i < len && arr[i][j] == 0) {  arr[i++][j] = m;  } else {  k = 3;  i--;  j--;  m--;  }  } else if (k == 3) {  if (j >= 0 && arr[i][j] == 0) {  arr[i][j--] = m;  } else {  k = 4;  i--;  j++;  m--;  }  } else if (k == 4) {  if (i >= 0 && arr[i][j] == 0) {  arr[i--][j] = m;  } else {  k = 1;  i++;  j++;  m--;  }  }  }  // 遍历  for (int m = 0; m < arr.length; m++) {  for (int n = 0; n < arr[m].length; n++) {  System.out.print(arr[m][n] + "\t");  }  System.out.println();  }  }  }  方式二：  class RectangleTest1 {  public static void main(String[] args) {  int n = 7;  int[][] arr = new int[n][n];  int count = 0; // 要显示的数据  int maxX = n - 1; // x轴的最大下标  int maxY = n - 1; // Y轴的最大下标  int minX = 0; // x轴的最小下标  int minY = 0; // Y轴的最小下标  while (minX <= maxX) {  for (int x = minX; x <= maxX; x++) {  arr[minY][x] = ++count;  }  minY++;  for (int y = minY; y <= maxY; y++) {  arr[y][maxX] = ++count;  }  maxX--;  for (int x = maxX; x >= minX; x--) {  arr[maxY][x] = ++count;  }  maxY--;  for (int y = maxY; y >= minY; y--) {  arr[y][minX] = ++count;  }  minX++;  }  for (int i = 0; i < arr.length; i++) {  for (int j = 0; j < arr.length; j++) {  String space = (arr[i][j] + "").length() == 1 ? "0" : "";  System.out.print(space + arr[i][j] + " ");  }  System.out.println();  }  }  } |

* 下面数组定义正确的有

A.String strs[] = { ‘a’ ‘b’ ‘c’};

B.String[] strs = {“a”, “b”, “c”};

C.String[] strs = new String{“a” ”b” ”c”};

D.String strs[] = new String[]{“a”, “b”, “c”};

E.String[] strs = new String[3]{“a”, “b”, “c”};

答案：BD

* 写出结果

|  |
| --- |
| class Demo{  public static void main(String[] args){  String foo="blue";  boolean[] bar=new boolean[2];  if(bar[0]){  foo="green";  }  System.out.println(foo);  }  } |

[输出]：blue

* 下面哪个数组定义是错误的, 并对错误的答案加上单行注释，写出错误的原因。

A, float[]=new float[3]; // 没有变量名

B, float f2[]=new float[];// 定义错误

C, float[] f1=new float[3];// 正确

D, boolean[] b={"true","false","true"};//错误, 不是字符串

E, double f4[]={1,3,5}; //正确

F, int f5[]=new int[3]{2,3,4}; //错误

G, float f4[]={1.2F,3.0,5.4};//错误, 不是float

* 练习：

1、从键盘输入本组学员的成绩，放到数组中

2、用for循环显示所有学员的成绩

3、排序：从低到高

4、查找是否有正好60分的，如果有返回位置

5、复制成绩最低三名构成新数组

6、用工具类打印成绩最低三名成绩

|  |
| --- |
| /\*  练习：  1、从键盘输入本组学员的成绩，放到数组中  2、用foreach显示所有学员的成绩  3、排序：从低到高  4、查找是否有正好60分的，如果有返回位置  5、复制成绩最低三名构成新数组  6、用工具类打印成绩最低三名成绩  \*/  import java.util.Scanner;  import java.util.Arrays;  class TestArraysExer{  public static void main(String[] args){  //1、声明一个数组并创建一个数组  int[] scores = new int[5];    //2、从键盘输入成绩  Scanner input = new Scanner(System.in);  for(int i=0; i<scores.length; i++){  //成绩存在数组的元素中  //为元素赋值  System.out.print("请输入第" + (i+1) + "个学员的成绩:");  scores[i] = input.nextInt();  }    //3、显示成绩  //用foreach显示所有学员的成绩  System.out.println("本组学员的成绩如下：");  for(int s = 0; s < scores.length;i++){  System.out.println(scores[s]);  }  //4、排序：从低到高  Arrays.sort(scores);    System.out.println("排序后的结果：" + Arrays.toString(scores));    //5、查找60分  int index = Arrays.binarySearch(scores, 60);  if(index<0){  System.out.println("没有正好60分的");  }else{  System.out.println("60分的索引位置：" + index);  }    //6、复制成绩最低三名构成新数组  //int[] newArray = Arrays.copyOfRange(scores, 0, 3);  int[] newArray = Arrays.copyOf(scores, 3);    //7、用工具类打印成绩最低三名成绩  System.out.println("成绩最低的三名同学是：" + Arrays.toString(newArray));  }  } |

* 创建一个char类型的26个元素的数组，分别 放置'A'-'Z'。使用for循环访问所有元素并打印出来。

|  |
| --- |
| Char [] arr = {‘A’,…..’Z’};  System.out.println(Arrays.toString(arr)); |

* 创建一个char类型的36个元素的数组，前26个元素放置'A'-'Z', 后10个元素放置'0'-'9'。使用for循环访问所有元素并打印出来。提示：char类型数据运算 'A'+1 -> 'B'，'0'+1 -> '1'

|  |
| --- |
| Char [] arr = {‘A’,…..’Z’,’0’….’9’};  System.out.println(Arrays.toString(arr)); |

* 一维数组a[]，用java代码将数组元素顺序颠倒

|  |
| --- |
| public static void main(String[] args) {  int[] a = new int[] { (int) (Math.random() \* 1000),(int) (Math.random() \* 1000), (int) (Math.random() \* 1000),(int) (Math.random() \* 1000), (int) (Math.random() \* 1000) };  System.out.println(a);  System.out.println(Arrays.toString(a));  swap(a);  System.out.println(Arrays.toString(a));  }  public static void swap(int a[]) {  int len = a.length;  for (int i = 0; i < len / 2; i++) {  int tmp = a[i];  a[i] = a[len - 1 - i];  a[len - 1 - i] = tmp;  }  } |

* 任意一个人输入年龄，就可以判断出他是哪个年龄段的人？

|  |
| --- |
| /\*  0-9 儿童  10-19 少年  20-29 青少年  30-39 青年  40-49 壮年  50-59 中年  60-69 中老年  70-79 老年  80-89 老老年  90-99 老老老年  \*/  //1.switch  int age = 34;  switch(age/10){  case 0: ....0-9....  case 1: ....10-19....  case 2: .....  }  //2.if...else.....  if(age>=0 && age<10){  .....  }else if(age>=10 && age< 20){  ....  }else......    //3.数据结构的思想  int age = 22;  String[] arr = {"儿童","少年","青少年","青年","壮年","中年","","","",""};  int idx = age/10;  System.out.println(arr[idx]); |

* 有一个已经排好序的数组。现输入一个数，要求按原来的规律将它插入数组中。 程序分析：首先判断此数是否大于最后一个数，然后再考虑插入中间的数的情况，插入后此元素之后的数，依次后移一个位置。

|  |
| --- |
| // 插入排序 |

* 输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组。

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Arrays;  /\*\*  \* 练习:输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组。  \*  \* @author hfm  \* @version 1.01 2020-03-19 19:34  \* @date 2020/3/19  \*/  public class ChangeArray {  public static void main(String[] args) {  int[] arr = new int[]{33, 12, 34, 45, 67, 53, 76, 34};  changeArr(arr);  }  private static void changeArr(int[] arr) {  int max = arr[0];  int min = arr[0];  // 获取最大及最小值  for (int i = 0; i < arr.length; i++) {  if (max < arr[i]) {  max = arr[i];  }  if (min > arr[i]) {  min = arr[i];  }  }  int minNum = 0;  int maxNum = 0;  // 获取最大值与最小值网址  for (int i = 0; i < arr.length; i++) {  if (arr[i] == max) {  maxNum = i;  }  if (arr[i] == min) {  minNum = i;  }  }  //System.out.println(maxNum+" "+minNum);  //System.out.println(max+" "+min);  int temp = 0;  temp = arr[0];  arr[0] = arr[maxNum];  arr[maxNum] = temp;  temp = arr[arr.length - 1];  arr[arr.length - 1] = arr[minNum];  arr[minNum] = temp;  System.out.println(Arrays.toString(arr));  }  } |

* 有n个整数，使其前面各数顺序向后移m个位置，最后m个数变成最前面的m个数

|  |
| --- |
| package com.hfm.baisc.java;  import java.util.Arrays;  import java.util.Scanner;  /\*\*  \* 练习:有n个整数，使其前面各数顺序向后移m个位置，最后m个数变成最前面的m个数  \*  \* @author hfm  \* @version 1.01 2020-03-19 19:49  \* @date 2020/3/19  \*/  public class MoveArray {  public static void main(String[] args) {  int[] arr = new int[]{343, 45, 232, 678, 8, 8644};  Scanner scanner = new Scanner(System.in);  System.out.println("请输入移动的个数");  int m = scanner.nextInt();  int[] temps1 = new int[m];  int count = 0;  // 将后M个数存储到temps1  for (int i = arr.length - m; i < arr.length; i++) {  temps1[count] = arr[i];  count++;  }  // 将前面的数存储到temps2  count = 0;  int[] temps2 = new int[arr.length - m];  for (int i = 0; i < arr.length - m; i++) {  temps2[count] = arr[i];  count++;  }  count =0;  for (int i = 0; i < arr.length; i++) {  if (i < m) {  arr[i] = temps1[i];  }else{  arr[i] = temps2[count];  count ++;  }  }  System.out.println(Arrays.toString(arr));  }  } |

* 定义一个4行4列的二维数组，逐个从键盘输入值，然后将第1行和第4行的数据进行交换，将第2行和第3行的数据进行交换

|  |
| --- |
| int[][] arr = new int[][]{{1, 1, 1, 1}, {2, 2, 2, 2}, {3, 3, 3, 3}, {4, 4, 4, 4}};  System.out.println(Arrays.deepToString(arr));  int temp = 0;  for (int i = 0; i < arr.length / 2; i++) {  for (int j = 0; j < arr[i].length; j++) {  temp = arr[i][j];  arr[i][j] = arr[arr.length -1- i][j];  arr[arr.length -1- i][j] = temp;  }  }  System.out.println(Arrays.deepToString(arr)); |

* 定义一个3行4列的二维数组，逐个从键盘输入值，编写程序将四周的数据清0

|  |
| --- |
| for (int i = 0; i < arr.length; i++) {  for (int j = 0; j < arr[i].length; j++) {  if (i == 0 || j == 0 || i == arr.length - 1 || j == arr[i].length - 1) {  arr[i][j] = 0;  }  }  }  System.out.println(Arrays.deepToString(arr)); |

* 定义一个3行4列的二维数组，逐个从键盘输入值，将左下三角的值清0

|  |
| --- |
| arr = new int[][]{{1, 1, 1, 1}, {2, 2, 2, 2}, {3, 3, 3, 3}, {4, 4, 4, 4}};  for (int i = 0; i < arr.length; i++) {  for (int j = 0; j < arr[i].length; j++) {  if (j < i) {  arr[i][j] = 0;  }  }  }  System.out.println(Arrays.deepToString(arr)); |

* 定义一个4行4列的二维数组，逐个从键盘输入值，将对角线的值清0

|  |
| --- |
| arr = new int[][]{{1, 1, 1, 1}, {2, 2, 2, 2}, {3, 3, 3, 3}, {4, 4, 4, 4}};  for (int i = 0; i < arr.length; i++) {  for (int j = 0; j < arr[i].length; j++) {  if (j == i) {  arr[i][j] = 0;  }  }  }  System.out.println(Arrays.deepToString(arr)); |

* 定义一个N\*N二维数组，从键盘上输入值，找出每行中最大值组成一个一维数组并输出

|  |
| --- |
| arr = new int[][]{{1, 1, 1, 1}, {2, 2, 2, 2}, {3, 3, 3, 3}, {4, 4, 4, 4}};  int [] newArr = new int[arr.length];  int max;  for (int i = 0; i < arr.length; i++) {  max = arr[i][0];  for (int j = 0; j < arr[i].length; j++) {  if(max < arr[i][j]){  max = arr[i][j];  }  }  newArr[i] = max;  }  System.out.println(Arrays.toString(newArr)); |

* 要求用户输入圆形的半径,求出圆形的面积和周长,并将结果显示在屏幕上.

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 要求用户输入圆形的半径,求出圆形的面积和周长,并将结果显示在屏幕上.  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class CircularCalculation {  public static void main(String[] args) {  // 创建键盘输入对象  Scanner scanner = new Scanner(System.in);  System.out.println("请输入圆的半径");  int radius = scanner.nextInt();  System.out.println("输入的半径为" + radius);  double pi = 3.14;  double perimeter = 2 \* radius \* pi;  System.out.println("圆的周长为" + perimeter);  double area = pi \* Math.pow(radius, 2);  System.out.println("圆的面积为" + area);  }  } |

* 判断一个整数是偶数还是奇数

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 联系 判断一个整数是偶数还是奇数  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class JudgingNumbers {  public static void main(String[] args) {  // 创建键盘输入对象  Scanner scanner = new Scanner(System.in);  System.out.println("请输入要判断的整数");  int num = scanner.nextInt();  if (num % 2 == 0) {  System.out.println("输入的" + num + "是偶数");  } else {  System.out.println("输入的" + num + "是奇数");  }  }  } |

* 读取文件

|  |
| --- |
| package cn.hfming.baiscjava;  import java.io.\*;  /\*\*  \* 练习 读取文件  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class FileRead {  public static void main(String[] args) {  // 创建文件对象  File file = new File("D:\\intelliJIDEAWorkplace\\core-java\\homework\\read.txt");  // 建立数据的输入通道  FileInputStream fileInputStream = null;  try {  fileInputStream = new FileInputStream(file);  // 创建缓存数组  byte[] content = new byte[1024];  // 循环读取数据  while ((fileInputStream.read(content)) != -1) {  // 输出读到的数据  System.out.println(new String(content, 0, 1024));  }  } catch (FileNotFoundException e) {  throw new RuntimeException(e);  } catch (IOException e) {  throw new RuntimeException(e);  } finally {  try {  fileInputStream.close();  } catch (IOException e) {  throw new RuntimeException(e);  }  }  }  } |

* 给定3个数a，b和c,输出最大的数。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 给定3个数a，b和c,输出最大的数。  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class GetBiggest {  public static void main(String[] args) {  // 创建键盘输入对象  Scanner scanner = new Scanner(System.in);  System.out.println("请依次输入三个数并用“，”号隔开");  String str = scanner.next();  // 将输入的字符串切割开  String[] arr = str.split(",");  int[] arr2 = new int[arr.length];  // 将字符串转发为数字  for (int i = 0; i < arr2.length; i++) {  arr2[i] = Integer.parseInt(arr[i]);  }  // 选出最大的数字  int max = arr2[0];  for (int i = 0; i < arr2.length; i++) {  if (arr2[i] > max) {  max = arr2[i];  }  }  System.out.println("输入的三个数字中最大的为" + max);  }  } |

* 某电信公司的市内通话费计算标准如下：三分钟内0.2元，三分钟后每增加一分钟增加0.1元，不足一分钟的按一分钟计算。要求编写程序，给定一个通话时间（单位：秒），计算出应收费金额。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  import java.util.ServiceConfigurationError;  /\*\*  \* 练习 某电信公司的市内通话费计算标准如下：三分钟内0.2元，三分钟后每增加一分钟增加0.1元，不足一分钟的按一分钟计算。要求编写程序，给定一个通话时间（单位：秒），计算出应收费金额。  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class Amount {  public static void main(String[] args) {  // 创建键盘输入对象  Scanner scanner = new Scanner(System.in);  System.out.println("请输入通话时间");  double time = scanner.nextDouble();  if(time <= 3.0){  System.out.println("收费0.2元");  }else if(time > 3){  System.out.println("收费" + (0.2 + Math.floor(time - 3) \* 0.1));  }  }  } |

* 某市的出租车计费标准为：3公里以内10元，3公里以后每加0.5公里加收1元；每等待2.5分钟加收1元；超过15公里的加收原价的50%为空驶费。要求编写程序，对于任意给定的里程数（单位：公里）和等待时间（单位：秒）计算出应付车费，车费直接截去小数位，只取整数。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 某市的出租车计费标准为：3公里以内10元，3公里以后每加0.5公里加收1元；每等待2.5分钟加收1元；超过15公里的加收原价的50%为空驶费。要求编写程序，对于任意给定的里程数（单位：公里）和等待时间（单位：秒）计算出应付车费，车费直接截去小数位，只取整数。  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class Amount2 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入里程数");  double mileage = scanner.nextDouble();  System.out.println("请输入等待时间");  double time = scanner.nextDouble();  if (mileage <= 3) {  System.out.println("收费" + Math.floor(time / 2.5 + 10));  } else if (mileage < 15 && mileage > 3) {  System.out.println("收费" + Math.floor(10 + (mileage - 3) / 0.5 + time / 2.5));  } else {  System.out.println("收费" + Math.floor((10 + (mileage - 3) / 0.5 + time / 2.5)) \* 1.5);  }  }  } |

* 编写程序，判断给定的某个年份是否是闰年。闰年的判断规则如下：（1）若某个年份能被4整除但不能被100整除，则是闰年。（2）若某个年份能被400整除，则也是闰年。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 编写程序，判断给定的某个年份是否是闰年。闰年的判断规则如下：（1）若某个年份能被4整除但不能被100整除，则是闰年。（2）若某个年份能被400整除，则也是闰年。  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class LeapYear {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入年份");  int year = scanner.nextInt();  if ((year % 100 != 0 && year % 4 == 0) || year % 400 == 0) {  System.out.println(year + "是闰年");  } else {  System.out.println(year + "不是闰年");  }  }  } |

* 给定一个百分制的分数，输出相应的等级。

90分以上 A级

80~89 B级

70~79 C级

60~69 D级

60分以下 E级

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 .给定一个百分制的分数，输出相应的等级。  \* 90分以上 A级  \* 80~89 B级  \* 70~79 C级  \* 60~69 D级  \* 60分以下 E级  \*  \* @author hfm  \* @version 1.01 2019-10-14  \*/  public class GetGrade {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入成绩");  double grade = scanner.nextDouble();  if (grade < 60 && grade >= 0) {  System.out.println("成绩为E");  }  if (grade >= 60 && grade < 70) {  System.out.println("成绩为D");  }  if (grade >= 70 && grade < 80) {  System.out.println("成绩为C");  }  if (grade >= 80 && grade < 90) {  System.out.println("成绩为B");  }  if (grade >= 90 && grade <= 100) {  System.out.println("成绩为A");  }  if (grade < 0 || grade > 100) {  System.out.println("输入错误");  }  }  } |

* 编写程序求 1+3+5+7+……+99 的和值。

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 编写程序求 1+3+5+7+……+99 的和值。  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetSum2 {  public static void main(String[] args) {  int sum = 0;  for (int i = 1; i <= 99; i++) {  sum += i;  }  System.out.println("1到99的和为" + sum);  }  } |

* 求所有满足如下条件的四位数：千位上的数字大于百位数字，百位数字大于十位数字，十位数字大于个位数字，并且千位数字是其他三位数字的和。

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 求所有满足如下条件的四位数：千位上的数字大于百位数字，百位数字大于十位数字，十位数字大于个位数字，并且千位数字是其他三位数字的和。  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetNum3 {  public static void main(String[] args) {  for (int i = 1000; i <= 9999; i++) {  // 千位  int a1 = i / 1000;  // 百位  int b1 = (i / 100) % 10;  // 十位  int c1 = (i / 10) % 100 % 10;  // 个位  int d1 = i % 10;  if (a1 > b1 && b1 > c1 && c1 > d1 && (a1 == b1 + c1 + d1)) {  System.out.print(i + " ");  } else {  continue;  }  }  }  } |

* 给定两个任意的正整数,求它们之间的所有完全平方数，完全平方数是指这个数是 某个整数的平方，例如 16，25，36等都是完全平方数。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 给定两个任意的正整数,求它们之间的所有完全平方数，完全平方数是指这个数是 某个整数的平方，例如 16，25，36等都是完全平方数。  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetNum4 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入两个正整数");  System.out.print("第一个正整数为：");  int a = scanner.nextInt();  System.out.print("第二个正整数数为：");  int b = scanner.nextInt();  //求最小值  int max;  int min;  if (a <= b) {  min = a;  max = b;  } else {  min = b;  max = a;  }  System.out.println(min);  System.out.println(max);  // 所有完全平方数  for (int i = min; i <= max; i++) {  if ((int) Math.pow((int) Math.sqrt(i), 2) == i) {  System.out.println(i);  }  }  }  } |

* 编写程序求下列多项式的前50项的和：1-1/3+1/5-1/7+1/9-……

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 编写程序求下列多项式的前50项的和：1-1/3+1/5-1/7+1/9-……  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetSum3 {  public static void main(String[] args) {  double result = 0;  for (int i = 1; i <= 50; i++) {  result += Math.pow(-1, i + 1) / (2 \* i - 1);  System.out.println(result);  }  System.out.println(result);  }  } |

* 请编写程序输出九九乘法表。

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 请编写程序输出九九乘法表  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class MultiplicationTable {  public static void main(String[] args) {  for (int i = 1; i <= 9; i++) {  for (int j = 1; j <= i; j++) {  System.out.print(i + "\*" + j + "=" + (i \* j) + "\t");  }  System.out.println();  }  }  } |

* 求斐波那契数列前n项的和值,斐波那契数列如下： 1，1，2，3，5，8，13，21……前两位数是1，从第三位开始每位数都是前两位数之和。

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 求斐波那契数列前n项的和值,斐波那契数列如下： 1，1，2，3，5，8，13，21……前两位数是1，从第三位开始每位数都是前两位数之和。  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class FbnqList {  public static void main(String[] args) {  //定义第一个加数a，初始值为1；定义第二个加数b，初始值为1；定义两个加数之和为c，初始值为0  int a = 1;  int b = 1;  int c = 0;  //首先在控制台打印出数列中第一个数和第二个数的值  System.out.print(a + "\t" + b + "\t");  //建立一个for循环，用于循环输出数列中第三位至第十位的数字  for (int i = 3; i <= 10; i++) {  //第三个数即为c，a+b等于c的值  c = a + b;  //将第一个加数a赋值为数列中的第二个数b的值  a = b;  //将第二个加数b赋值为数列中的第三个数c的值  b = c;  //在第二次循环打印时，将打印数列中的第四个数为：b + c = b + (a + b)  System.out.print(c + "\t");  }  }  } |

* 给定一个整数，把它的各位数字倒排过来形成一个新的整数例如：给定12345 输出54321 ；给定 9870 输出789

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 给定一个整数，把它的各位数字倒排过来形成一个新的整数例如：给定12345 输出54321 ；给定 9870 输出789  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetNum5 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入一个整数");  int input = scanner.nextInt();  int m = 0;  while (input > 0) {  m \*= 10;  m += input % 10;  input /= 10;  }  System.out.println(m);  }  } |

* 如何理解函数（方法）？

函数是对实现某一功能代码的封装，函数的实现有利于功能代码的重复使用。

* 怎么定义一个函数？

public(访问权限修饰符) static void/返回值类型 函数名(型参列表){

函数体；

return 返回值；//用于停止函数

}

* 函数重载什么时候使用？

当前的代码不能满足需求时需要对函数进行重载。

* 编写一个方法接受一个用户名与密码，如果输入的用户名不是admin密码不是888，那么继续提示用户“用户名或者密码错误”，否则提示用户输入正确。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Scanner;  /\*\*  \* 练习 编写一个方法接受一个用户名与密码，如果输入的用户名不是admin密码不是888，那么继续提示用户“用户名或者密码错误”，否则提示用户输入正确。  \* @author hfmn  \* @version 1.01 2019-10-15  \*/  public class Admin {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入用户名");  String name = scanner.next();  System.out.println("请输入密码");  String scrit = scanner.next();  if (("admin").equals(name) && ("888").equals(scrit)){  System.out.println("登陆成功");  }else{  System.out.println("用户名或密码错误");  }  }  } |

* 数组是什么和数组的特点是什么？

数组是存储一组相同数据的容器。

数组一旦第定义它的长度不会发生改变的。

数组中的元素都有下角标值，可以通过下角标值进行访问。

数组中元素与元素之间的内存地址是连续的。

* 对于一个int数组，把其中的所有元素都向后移动一个位置定义一个长度为10的int数组,统计数组的最大值、最小值、奇数和偶数的个数比如： int[] arr = { 1, 3, 5, 7, 9 } --> { 9, 1, 3, 5, 7 }

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 对于一个int数组，把其中的所有元素都向后移动一个位置定义一个长度为10的int数组,统计数组的最大值、最小值、奇数和偶数的个数  \* 比如： int[] arr = { 1, 3, 5, 7, 9 } --> { 9, 1, 3, 5, 7 }  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetArray {  public static void main(String[] args) {  int[] arr = new int[]{1,23,4,34356,5675};  int max = arr[0];  int min = arr[0];  int oddNum = 0;  int evenNum = 0;  for (int i = 0; i < arr.length; i++) {  if (max < arr[i]) {  max = arr[i];  }  }  System.out.println("最大值为：" + max);  for (int i = 0; i < arr.length; i++) {  if (min > arr[i]) {  min = arr[i];  }  }  System.out.println("最小值为：" + min);  for (int i = 0; i < arr.length; i++) {  if (arr[i] % 2 == 1) {  oddNum++;  }  }  System.out.println("奇数个数为:" + oddNum);  for (int i = 0; i < arr.length; i++) {  if (arr[i] % 2 == 0) {  evenNum++;  }  }  System.out.println("偶数个数为:" + evenNum);  }  } |

* 请编写1个程序 求出1个整型数组中的最大值和次大值。

|  |
| --- |
| package cn.hfming.baiscjava;  /\*\*  \* 练习 请编写1个程序 求出1个整型数组中的最大值和次大值。  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class GetMax {  public static void main(String[] args) {  int[] arr = new int[]{1,4345,56,56,43,454};  int max = 0;  int nex = 0;  for (int i = 0; i < arr.length; i++) {  if (max < arr[i]) {  max = arr[i];  }  }  System.out.println("最大值为：" + max);  for (int i = 0; i < arr.length; i++) {  if (nex < arr[i] && max > arr[i]) {  nex = arr[i];  }  }  System.out.println("次大值为：" + nex);  }  } |

* 用户输入班级的人数,然后依次输入每个同学的成绩. 输入完毕之后,如果及格率没有达到60%， 就为每1个没有及格的成绩加2分,直到及格率达到60%为止。

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Arrays;  import java.util.Scanner;  /\*\*  \* 练习 用户输入班级的人数,然后依次输入每个同学的成绩. 输入完毕之后,如果及格率没有达到60%， 就为每1个没有及格的成绩加2分,直到及格率达到60%为止。  \*  \* @author hfm  \* @version 1.01 2019.10.15  \*/  public class InproveGread {  public static void main(String[] args) {  System.out.println("请输入班级人数");  Scanner scanner = new Scanner(System.in);  int num = scanner.nextInt();  System.out.println("输入的班级人数为：" + num);  int[] arr = new int[num];  System.out.println("请依次输入班级同学的成绩");  for (int i = 0; i < arr.length; i++) {  arr[i] = scanner.nextInt();  }  System.out.println("正在计算平均分！");  int sum;  int average;  OUT:  for (; ; ) {  sum = 0;  average = 0;  for (int i = 0; i < arr.length; i++) {  sum += arr[i];  }  //System.out.println(sum);  average = sum / arr.length;  System.out.println(average);  if (average >= 60) {  System.out.println(Arrays.toString(arr));  break OUT;  } else {  for (int i = 0; i < arr.length; i++) {  if (arr[i] < 60) {  arr[i] += 2;  } else {  continue;  }  }  }  }  }  } |

* 现在有如下的一个数组：int oldArr[]={1,3,4,5,0,0,6,6,0,5,4,7,6,7,0,5} ，要求清除数组中为0的元素，然后存储非零的数据存储到一个新数组中。

|  |
| --- |
| package cn.hfming.baiscjava;  import cn.hfming.array.ArrayRemoveZero;  import java.util.Arrays;  import java.util.Scanner;  /\*\*  \* 练习 现在有如下的一个数组：int oldArr[]={1,3,4,5,0,0,6,6,0,5,4,7,6,7,0,5} ，要求清除数组中为0的元素，然后存储非零的数据存储到一个新数组中。  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class RemoveZero {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入数组，数值之间用“，”号隔开");  String arrString = scanner.next();  // 切割字符串  String[] arr = arrString.split(",");  int[] finalArr = new int[arr.length];  for (int i = 0; i < arr.length; i++) {  finalArr[i] = Integer.parseInt(arr[i]);  }  System.out.println(Arrays.toString(ArrayRemoveZero.arrayRemoveZreo(finalArr)));  }  }  package cn.hfming.array;  /\*\*  \* 数组工具类  \* 功能：除去数组中为0的元素，返回新的数组  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class ArrayRemoveZero {  public static int[] arrayRemoveZreo(int[] arr) {  int zeroNumber = 0;  for (int i = 0; i < arr.length; i++) {  if (arr[i] == 0) {  zeroNumber++;  }  }  int index = 0;  int[] newArray = new int[arr.length - zeroNumber];  for (int i = 0; i < arr.length; i++) {  if (arr[i] != 0) {  newArray[index] = arr[i];  index++;  }  }  return newArray;  }  } |

* 清除重复元素。 int[] arr = {1,4，9,4,1,1,7}；

|  |
| --- |
| package cn.hfming.baiscjava;  import java.util.Arrays;  import java.util.Scanner;  import cn.hfming.array.\*;  /\*\*  \* 练习 清除重复元素。 int[] arr = {1,4，9,4,1,1,7}；  \*  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class CleanRepeat {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.println("请输入数组，数值之间用“，”号隔开");  String arrString = scanner.next();  // 切割字符串  String[] arr = arrString.split(",");  int[] finalArr = new int[arr.length];  for (int i = 0; i < arr.length; i++) {  finalArr[i] = Integer.parseInt(arr[i]);  }  System.out.println(Arrays.toString(ArrayCleanRepeat.arrayCleanRepeat(finalArr)));  }  }  package cn.hfming.array;  /\*\*  \* 数组工具类  \* 功能：去除重复元素，返回新的数组  \* @author hfm  \* @version 1.01 2019-10-15  \*/  public class ArrayCleanRepeat {  public static int[] arrayCleanRepeat(int[] arr) {  // 统计重复元素的个数  int repeatNumber = 0;  for (int i = 0; i < arr.length; i++) {  for (int j = i + 1; j < arr.length; j++) {  if (arr[i] == arr[j]) {  repeatNumber++;  break;  }  }  }  //创建新的数组  int[] arr2 = new int[arr.length - repeatNumber];  int index = 0;  OUT:  for (int i = 0; i < arr.length; i++) {  for (int j = 0; j < arr2.length; j++) {  if (arr2[j] == arr[i]) {  continue OUT;  }  }  arr2[index] = arr[i];  index++;  }  return arr2;  }  } |

* 一台完整的计算机包括了（ ）

A. 计算机及外部设备 B. 主机、显示器 C. 硬件系统与系统软件 D. windows操作系统

C 计算机: 硬件 + 软件 = 计算机

* 计算机存储数据的最小单位是二进制的( )。

A.位（比特） B.字节 C.字长 D.千字节KB

B

位：最小的存储单位称为位（bit）。只能容纳两个值（0或1）之一，不能在一个位中存储更多的信息。位是计算机存储的基本单位。位表示的是二进制位，一般称为比特，是计算机存储的最小单位，类似于（两，公斤，克）的概念，比如：11010100是一个8位二进制数。一个二进制位只可以表示0和1两种状态；两个二进制位可以表示00、01、10、11四种（22）状态；三位二进制数可表示八种状态。

字节：字节(byte)是常用的计算机存储单位。字节的标准定义：一个字节均为8位。由于上述所讲每个位或者是0或者是1，所以一个8位的字节包含256种可能的0，1组合。字节来自英文Byte，音译为“拜特”，习惯上用大写的“B”表示。字节是计算机中数据处理的基本单位。计算机中以字节为单位存储和解释信息，规定一个字节由八个二进制位构成，即1个字节等于8个比特（1Byte=8bit）。八位二进制数最小为00000000，最大为11111111；通常1个字节可以存入一个ASCII码，2个字节可以存放一个汉字国标码

字符：对于一种给定的计算机设计，字(word)是自然的存储单位。一个英文字符和英文标点占用一个字节，一个中文字符和中文标点占用2个字节。计算机进行数据处理时，一次存取、加工和传送的数据长度称为字（word）。一个字通常由一个或多个（一般是字节的整数位）字节构成。例如286微机的字由2个字节组成，它的字长为16；486微机的字由4个字节组成，它的字长为32位机。计算机的字长决定了其CPU一次操作处理实际位数的多少，由此可见计算机的字长越大，其性能越优越。

字长：计算机的每个字所包含的位数称为字长。根据计算机的不同，字长有固定的和可变的两种。固定字长，即字长度不论什么情况都是固定不变的；可变字长，则在一定范围内，其长度是可变的。计算的字长是指它一次可处理的二进创数字的数目。计算机处理数据的速率，自然和它一次能加工的位数以及进行运算的快慢有关。如果一台计算机的字长是另一台计算机的两倍，即使两台计算机的速度相同，在相同的时间内，前者能做的工作是后者的两倍。一般地，大型计算机的字长为32―64位，小型计算机为12―32位，而微型计算机为4一16位。字长是衡量计算机性能的一个重要因素。

bps：bits per second 的简称。一般数据机及网络通讯的传输速率都是以「bps」为单位。如56Kbps、100.0Mbps 等等。

Bps：Byte per second 的简称。而电脑一般都以Bps 显示速度，如1Mbps 大约等同 128 KBps。

bit：电脑记忆体中最小的单位，在二进位电脑系统中，每一bit 可以代表0 或 1 的数位讯号。

Byte：一个Byte由8 bits 所组成，可代表一个字元(A~Z)、数字(0~9)、或符号(,.?!%&+-\*/)，是记忆体储存资料的基本单位。

至於每个中文字则须要两Bytes。当记忆体容量过大时，位元组这个单位就不够用，因此就有千位元组的单位KB出现，以下乃个记忆体计算单位之间的相关性：

1 Byte = 8 Bits

1 KB = 1024 Bytes

1 MB = 1024 KB

1 GB = 1024 MB

usb2.0标准接口传输速率。许多人都将“480mbps”误解为480兆/秒。其实，这是错误的，事实上“480mbps”应为“480兆比特/秒”或“480兆位/秒”，它等于“60兆字节/秒”，

* 以下不是操作系统的是（ C ）

A. Windows B. Linux C. Java D. Android

* 操作系统中的文件名中不允许使用( C )

A. 数字（0～9） B. 美元符号（$） C. 反斜线（\） D. 小写英文字母（a～z）

DOS操作系统规定文件名由文件主名和扩展名组成，文件主名由1~8个字符组成，扩展名由1~3个字符组成，主名和扩展名之间由一个小圆点隔开，一般称为8.3规则。其格式如下：

例如： G9401.DBF，这里G9401是主名，DBF是扩展名。文件主名和扩展名可以使用的字符是：

（1）英文字母：A—z（大小写等价）

（2）数 字：0—9

（3）汉 字

（4）特殊符号：$#&@（）-[]^～等。

（5）各种控制字符和下列字符不能用在文件名中：< > / \ | : " \* ?

因为这些字符已做它用。

windows突破了DOS对文件命名规则的限制，允许使用长文件名，其主要命名规则如下：

（1）文件名最长可以使用255个字符。

（2）可以使用扩展名，扩展名用来表示文件类型，也可以使用多间隔符的扩展名。如win.ini.txt是一个合法的文件名，但其文件类型由最后一个扩展名决定。

（3）文件名中允许使用空格，但不允许使用下列字符（英文输入法状态）：< > / \ | : " \* ?

（4）windows系统对文件名中字母的大小写在显示时有不同，但在使用时不区分大小写。

文件名是文件存在的标识,操作系统根据文件名来对其进行控制和管理.不同的操作系统对文件命名的规则略有不同,即文件名的格式和长度因系统而异.例如:UNIX/LINUX系统区分文件名的大小写,而有些系统则不区分如:MS-DOS和WINDOWS.

对于文件扩展名是用来区分文件的属性的.在有的系统中并不为系统所感知如:UNIX/LINUX系统中,hello.c文件与helloc文件对于系统来说没有什么区别,加上了扩展名.c与不加都一样,系统都不知道那是个什么文件.只是用户加上.c后,方便用户明白那是c语言源码文件.但是在MS-DOS与WINDOWS系统中,文件扩展名是为系统感知的.在WINDOWS中,对于不同的文件扩展名有其相关联的程序进行打开,这些扩展名除了如txt等是原来就有的以外其余一般是安装软件时被注册到系统中的.在注册表中的HKEY-CLASSES-ROOT根键下可以看到其相关信息.若将文件扩展名改为系统不认知的扩展名,则在打开时提示选择其打开程序.如将hello.txt改为hello.123,同时可以看到图标也发生了变化,变为不可知.但是对于如.doc格式的WORD文件,由于是MICROSOFT开发的,保持了很好的兼容性,系统在感知时并不只检查扩展名,还检查其文件内部格式标识,所以即使将.doc改为其它字符,在装了WORD的系统中,系统仍能够使其关联到WORD进行打开.

* Java是什么( A )

A.计算机语言 B. 操作系统 C. 计算机单位 D. 操作系统的工具

* 运行java程序的时候是至少要安装什么（ C ）

A. JDK B. Office办公软件 C. JRE D. C语言的程序

JDK是整个Java的核心，包括了Java运行环境、Java工具和Java基础类库。

JDK: java开发工具集

JDK = java开发工具 + jre

JRE: java运行环境。

JRE = java虚拟机 + 核心类库(辅助java虚拟机运行的文件)

JRE(Java Runtime Environment)Java运行环境。包括Java虚拟机(JVM Java Virtual Machine)和Java程序所需的核心类库等，如果想要运行一个开发好的Java程序，计算机中只需要安装JRE即可。

* Java程序的执行过程中用到一套JDK工具，其中javac.exe指（ A ）

A. Java编译器 B. Java字节码解释器 C. Java文档生成器 D. Java类分解器

javac.exe: 启动java编译器对java源文件进行编译。

java.exe: 启动java虚拟机解释并执行指定的class文件。

* Java源文件和编译后的文件扩展名分别为（ B ）

A .class和.java B .java和.class C .class和.class D.java和.java

* 阅读以下程序判断会输出几次”hello world”( B )

int num = 0;

while(num < 5){

System.out.println("Hello World!");

}

A. 1次 B. 5次 C. 0次 D. 无数次

* 15二进制数是多少（ C ）

A. 00000111 B. 00000011 C. 00001111 D. 00011101

15/2 ... 1

7/2 ... 1

3/2 ... 1

1/2 ... 1

1111

* 请你仔细观察数列的排列规律，选出括号中应写的数字：（ C ）

18，-27，36，( )，54

A.44 B．45 C．-45 D．-44

* 5000秒是几小时几分钟几秒？（需要写出计算过程）

5000/60/60=1小时

5000-3600=1400s/60=23分钟

1400-1380=20秒

* 根据下面的条件算出这个人生于何年，死于何年？（需要写出计算过程）

1.他生于19世纪，也同样死于19世纪；

2.他诞生的年份和逝世的年份都是由4个相同的数字组成，但排列的位置不同；

3.他诞生的那一年，4个数字之和是14；

4.他逝世那一年的数字的十位数是个位数的4倍。

18 小于100

18xy 18yx

1+8+x+y=14

Y=4x

X=1,y=4

生于1814，死于1841

* 计算兔子的繁殖数量（需要写出计算过程）

兔子的繁殖能力很强，一对兔子每个月可以生一对小兔子，而一对小兔子生下后第二个月也开始生小兔子。如果从刚生的一对兔子算起，那么一年可以繁殖出多少对兔子。(提示：斐波那契数列)

|  |
| --- |
| public static void pratice1() {  int result = 1;  // 记录上一个result  int x = 0;  // 记录上上个result  int y = 0;  // 保存上一个x的值  int temp = 0;  for (int i = 1; i <= 12; i++) {  y = temp;  if (i == 1 || i == 2) {  result = 1;  } else {  result = x + y;  }  temp = x;  // 将result赋值给x,让x记录上一个result  x = result;  }  System.out.println(result);  }  //定义第一个加数a，初始值为1；定义第二个加数b，初始值为1；定义两个加数之和为c，初始值为0  int a = 1;  int b = 1;  int c = 0;  //首先在控制台打印出数列中第一个数和第二个数的值  System.out.print(a + "\t" + b + "\t");  //建立一个for循环，用于循环输出数列中第三位至第十位的数字  for (int i = 3; i <= 10; i++) {  //第三个数即为c，a+b等于c的值  c = a + b;  //将第一个加数a赋值为数列中的第二个数b的值  a = b;  //将第二个加数b赋值为数列中的第三个数c的值  b = c;  //在第二次循环打印时，将打印数列中的第四个数为：b + c = b + (a + b)  System.out.print(c + "\t");  } |

* 在控制台输出“Hello world.” 请写出相应的Java代码

|  |
| --- |
| class HelloWorld {  public static void main(String[] args) {  System.out.println("HelloWorld");  }  } |

* 下列哪个声明是错误的？（b）

A. int i=10;

B. float f=1.1; //值需要加f

C. double d=34.4;

D. byte b=127;

* 8种基本数据类型byte 、 short 、 int 、 long 、 float 、 double 、 char 、 Boolean

注意: 一个小数如果没有加上任何的标识的时候，默认都double类型。如果是float类型的数据那么该数据必须要以f结尾。不区分大小写。

注意: 一个整数没有加上任何的标识的时候，那么该整数默认是int类型 的数据。

如果一个数据需要表示成一个长整型，那么该数据要以L结尾。L是不区分大小写的，但是建议使用大写。

* 下面哪个不是java中的关键字？（c）

A. public

B. true

C. main //main是标识符

D. class

* 下面哪个语句不会产生编译错误？（C）

A. float a =2.0; //值需要加f

B. char c =”a”; //字符用''

C. byte b =25; //先判断25是否超出-128~127，超出才报错

D. boolean d=0; //只有两个值turn falst

byte(字节) 8bit

short(短整型) 16bit

int(整型) 32bit

long (长整型) 64bit

小数：

float(单精度浮点型) 32bit

double(双精度浮点型) 64bit

布尔：

Boolean(不利恩) 一个字节或者是 4 个字节。 只有两个值true ，false。

字符：

char 2 个字节

* 下面程序哪个语句是正确的（C）

A. byte a=0,b=3; byte c =a+b; //a+b是int类型

B. short s =23; s=s+12; //s+12是Int类型

C. short s=23; s+=12; //s+=12本身就是short类型

D. float f = 23+23.23; //23+23.23是Int类型

考察表达式的数据类型自动提升

所有的 byte 型、short 型和 char 的值将被提升到 int 型。

如果一个操作数是 long 型，计算结果就是 long 型。

如果一个操作数是 float 型，计算结果就是 float 型。

如果一个操作数是 double 型，计算结果就是 double 型。

对于 short s1 = 1; s1 = s1 + 1; 由于 s1+1 运算时会自动提升表达式的类型，所以结果是 int 型，再赋值给short 类型 s1 时，编译器将报告需要强制转换类型的错误。对于 short s1 = 1; s1 += 1;由于 += 是 java 语言规定的运算符，java 编译器会对它进行特殊处理，因此可以正确编译。

* byte类型的大小是（A）

A. -128 ~ 127

B. -228 ~128

C. -255 ~ 256

D. -255 ~ 255

考察基本数据类型使用的数据范围

整数：

byte(字节) 8bit

short(短整型) 16bit

int(整型) 32bit

long (长整型) 64bit

小数：

float(单精度浮点型) 32bit

double(双精度浮点型) 64bit

布尔：

Boolean(不利恩) 一个字节或者是 4 个字节。 只有两个值true ，false。

字符：

char 2 个字节

* 下面程序执行的结果是？（B）

public class Test{

public static void main(String[] args){

System.out.println(“”+’a’+1);

}

}

A. 98

B. a1

C. 971

D. 197

考察表达式的数据类型自动提升

所有数值运算符都可以用在 char 型数据上，如果另一个操作数是一个数字或者字符，那么 char 会自动提升为 int 型，如果另一个操作数是字符串，那么字符就会和字符串相连。

* 下面程序执行的结果是？（A）

int i = 100;

while(true){

if ( i++ > 100 ) //先判断，后++

break;

System.out.println(i);

}

A. 100

B. 101

C. 102

D. 103

考察while循环的使用 0到100共101个

* while的结构

while( 条件表达式 ){

执行语句；

}

* while循环语句要注意的细节

while 循环语句的循环次数一般是通过变量去控制的。

while 循环语句后面如果加上了分号，那么会影响循环的效果。

如果循环语句只有一个语句的时候，可以省略大括号，但是不建议省略。

* 下面程序执行的结果是？（C）

int a=2;

switch(a){

case 1:a+=1;break;

case 2:a+=2;

case 3:a+=3;break;

case 4:a+=4;break;

default:a=0;

}

System.out.println(a);

A. 5

B. 6

C. 7 //a=2+2+3

D. 8

考察switch选择判断语句的使用

1.switch 语句选择的类型只有四种：byte，short，int ， char。JDK1.7添加了String数据类型

2.case 之间与 default 没有顺序。先判断所有的 case，没有匹配的 case 执行 default。

3.switch 语句停止的条件是遇到了 break 关键字或者结束 switch 语句的大括号。

4.如果匹配的 case 或者 default 没有对应的 break，那么程序会继续向下执行，运行可以执行的语句，直到遇到 break 或者 switch 结尾结束。

5.switch case 中的值必须要与 switch 表达式的值具有相同的数据类型。而且 case 后跟的值必须是常量，不能跟变量。case就像选择题的答案之一。 break就是如果该答案正确那么就可以跳出switch了,意思就是说 已经找出了正确的答案了。那么这道题也就做完了。如果 case没有匹配接着进行下一个 case 匹配,直到匹配为止。 最后如果都没有匹配上,那么 switch 给提供了一个默认的答案,就是 default。

注意:case后跟的是冒号:每个case中的执行语句一定要加 break；

* 下面程序的运行结果是（d）

int a=3, b=1;

if(a==b)

System.out.println("a="+a);

A. a=1

B. a=3

C. 编译错误

D. 正常运行但没有输出

考察if条件语句的运用

忘记必要的括号：如果代码块中只有一条语句的时候，可以省略花括号，但是当花括号将多条语句扩在一起时，花括号就不能在省略。

double radius = 4;

double area;

if (radius >= 0)

area = radius \* radius \* 3.14;

System.out.println("The area " + " is " + area);

double radius = 4;

double area;

if (radius >= 0) {

area = radius \* radius \* 3.14;

System.out.println("The area " + " is " + area);

}

虽然代码一样多，但是第一个会编译报错（area 没有出初始化），第二个正常运行。就是因为少了花括号。所以一定要仔细

if 语句后出现分号

double radius = 0;

double area;

if (radius > 0); {

area = radius \* radius \* 3.14;

System.out.println("The area " + " is " + area);

}

注意：这是一个逻辑错误，编译和运行都不会报错，只是不会出现想要的结果。

相当于判断符合条件后，执行一个空语句。

double radius = 0;

double area;

if (radius > 0){}{

area = radius \* radius \* 3.14;

System.out.println("The area " + " is " + area);

}

* 下面程序的运行后,a,b,c的值正确的是：（b）

int a=1,b=2;

int c=(a+b>3?a++:++b);

A. a=2,b=3

B. a=1,b=3

C. a=1,b=2

D. c=2

考察三目运算符的使用

格式

(条件表达式)?表达式 1：表达式 2；

如果条件为 true，运算后的结果是表达式 1；

如果条件为 false，运算后的结果是表达式 2；

* 下面程序的运行结果（b）

public class Demo{

public static int fun(int c){

return c += 2;

}

public static void main(String[] args) {

int temp=fun(2);

System.out.println(temp);

}

}

A. 2

B. 4

C. 6

D. 8

运算符 运算 范例 结果

= 赋值 a=3,b=2 a=3,b=2

+= 加等于 a=3,b=3;a+=b; a=5,b=2;

-= 减等于 a=3,b=2,a-=b; a=1,b=2;

\*= 乘等于 a=3,b=2,a\*=b; a=6,b=2

/= 除等于 a=3,b=2,a/=b; a=1,b=2;

%= 模等于 a=3,b=2,a%=b; a=1,b=2

* 下面程序的运行结果，哪个是正确的（b）

int b=1;

while(++b<3)

System.out.println("LOOP");

A. 程序将会进入死循环导致无输出

B. 输出一次LOOP

C. 会输出多次LOOP

D. 程序中含有编译错误

考察while语句的使用及++自增运算符的使用

6．++(自增): 自增就是操作数+1

前自增: ++位于操作数的前面。 先自增，后使用。

后自增: ++位于操作数的后面。 先使用，后自增。

1先自增到2，输出一次，2先自增到3，不输出，总共输出一次

while 循环语句的循环次数一般是通过变量去控制的。

while 循环语句后面如果加上了分号，那么会影响循环的效果。

如果循环语句只有一个语句的时候，可以省略大括号，但是不建议省略。

* 下面数组定义错误的是（D）

A. int [] arr ={23,45,65,78,89};

B. int [] arr=new int[10] ;

C. int [] arr=new int[4]{3,4,5,6};

D. int [] arr={‘a’, 23 , 45 , 6}； //不同类型，不可以放在同一个数组

考察数组的定义

数组的定义格式

格式一：

数据类型[] 变量名 = new 数据类型[容量大小];

int [] arr = new int[8];

int: 表示该容器只能存储 int 类型的数据。

[] : 表示是一个数组类型。容器使用数组，那么如何来标识数组？.那么用一个特殊的符号[]中括号来表示。

arr： 变量名。

右边： new int[125]; 创建了一个 int 类型的数组对象，该对象的容量是 125 个数据。

new ： new 是用于创建一个对象的关键字。

int : 表示该数组对象只能存储 int 类型的数据

[]： 表示是一个数组类型。

8: 表示该数组容器的容量大小。

格式二：

元素类型[] 数组名 = new 元素类型[]{元素，元素，……};

int[] arr = new int[]{3,5,1,7};

int[] arr = {3,5,1,7};

* 下面程序执行的结果是？（D）

|  |
| --- |
| int x =1,y=1;  if(x++==2 & ++y==2){  x=7;  }  System.out.println("x="+x+" , y="+y); |

A. x=1 y=2

B. x=7 y=1

C. x=7 y=2

D. x=2 y=2

考察自增运算符

++(自增): 自增就是操作数+1

前自增: ++位于操作数的前面。 先自增，后使用。

后自增: ++位于操作数的后面。 先使用，后自增。

--（自减）:自减就是操作数-1

前自减：--位于操作数 的前面，先自减，后使用。

后自减: --位于操作数 的后面，先使用，后自减

如果运算符在变量的前面，则该变量自增 1 或者自减 1，然后返回的是变量的新值，如果运算符在变量的后面，则变量也会自增或者自减 1，但是返回的是变量原来的值。

++在前就是先运算，再取值，++在后就是先取值，再运算。

* 下面不属于基本数据类型的是（D）

A. int

B. double

C. long

D. int[] //数组是引用数据类型

考察基本数据类型

8种基本数据类型byte 、 short 、 int 、 long 、 float 、 double 、 char 、 Boolean

* 给出下列代码:

boolean a = false;

boolean b = true;

boolean c= ( a&&b)&&(!b);

int result = (c = = false)?1:2;

执行完后,c 与 result 的值是( A).

A. false和1

B. true和2

C. true和1

D. false和2

考察操作符的使用

“&”和“&&”的区别：单与时，左边无论真假，右边都进行运算；双与时，如果左边为真，右边参与运算，如果左边为假，那么右边不参与运算。

格式

(条件表达式)?表达式 1：表达式 2；

如果条件为 true，运算后的结果是表达式 1；

如果条件为 false，运算后的结果是表达式 2；

* 阅读下列代码

|  |
| --- |
| public class Test{  public static void main ( String []args ){  　　 int f=12;  　　 int i = 3;  　　 System.out.println( f / i );  　　}  }  } |

程序运行结果为( C).

A. 3

B. 3.0

C. 4

D. 4.0

* 下面程序执行的结果是？（ C）

|  |
| --- |
| boolean b=true;  if(b=false){  System.out.println("a");  }else if(b){  System.out.println(b);  }else if(!b){  System.out.println("c");  }else  System.out.println("d"); |

A. a

B. true

C. c

D. d

考察if条件语句的使用与赋值语句的使用

运算符 运算 范例 结果

= 赋值 a=3,b=2 a=3,b=2

+= 加等于 a=3,b=3;a+=b; a=5,b=2;

-= 减等于 a=3,b=2,a-=b; a=1,b=2;

\*= 乘等于 a=3,b=2,a\*=b; a=6,b=2

/= 除等于 a=3,b=2,a/=b; a=1,b=2;

%= 模等于 a=3,b=2,a%=b; a=1,b=2

* 下面程序执行的结果是？（D）

|  |
| --- |
| int x=2,y=3;  switch(x){  default:y++;  case 3:y++;  case 4:y++;  }  System.out.println("y="+y); |

A. 3

B. 4

C. 5

D. 6

考察switch选择判断语句的使用

1.switch 语句选择的类型只有四种：byte，short，int ， char。JDK1.7添加了String数据类型

2.case 之间与 default 没有顺序。先判断所有的 case，没有匹配的 case 执行 default。

3.switch 语句停止的条件是遇到了 break 关键字或者结束 switch 语句的大括号。

4.如果匹配的 case 或者 default 没有对应的 break，那么程序会继续向下执行，运行可以执行的语句，直到遇到 break 或者 switch 结尾结束。

5.switch case 中的值必须要与 switch 表达式的值具有相同的数据类型。而且 case 后跟的值必须是常量，不能跟变量。case就像选择题的答案之一。 break就是如果该答案正确那么就可以跳出switch了,意思就是说 已经找出了正确的答案了。那么这道题也就做完了。如果 case没有匹配接着进行下一个 case 匹配,直到匹配为止。 最后如果都没有匹配上,那么 switch 给提供了一个默认的答案,就是 default。

注意:case后跟的是冒号:每个case中的执行语句一定要加 break；

* 下面程序执行的结果

|  |
| --- |
| for(int i=1;i<=10;i++){  if (i<5)  continue;  System.out.println(“java基础班”);  } |

在屏幕上打印几次java基础班?(B )

A. 5

B. 6

C. 7

D. 8

考察循环语句中的continue关键字的作用

continue 的适用范围是循环语句中。

continue 关键字的作用： 跳过本次的循环体语句，继续下一次循环。

continue 关键字要注意的细节：

continue 关键字也可以配合标签使用。

continue 关键字后面也是不能直接跟语句的。

continue 关键字：语句将控制权传递给它所在的封闭迭代语句的下一次迭代。(跳出本循环，执行下一次循环)。

* 阅读下面代码段

|  |
| --- |
| public class Demo{  　　public static void main (String[] args){  int[] arr = new int [10];  　　 System.out.println(arr[1]);  　　}  } |

执行结果正确的说法是( C )

A. 编译时将产生错误

B. 编译时正确，运行时将产生错误

C. 输出0

D. 输出空

考察数组元素的初始值

* 和下面代码能完成相同的选项是（B）

|  |
| --- |
| int i=1;  int sum=0;  while(i<=100){  if(i%2==0){  sum=sum+i;  }  i++;  } |

A. for (int x =1; x<=100;x++){ sum=sum+x;}

B. for (int x =0; x<=100;x+=2){ sum=sum+x;}

C. for (int x =1; x<=100;x+=2){ sum=sum+x;}

D.上述全对

考察取模运算

5.%（取模，取余数） 取模运算的结果正负是取决于被除数的正负。

* 阅读下面代码段

|  |
| --- |
| public class Test{  　　public static void main (String[ ] args){  　　 char ch=’c’;  　　 switch (ch){  　　 case ‘a’:System.out.print(“a”);break;  　　 case ’b’: System.out.print(“ab”);  　　 case ’c’:System.out.print(“c”);  　　 default:System.out.print(“d”);  　　 }  　　}  } |

输出的结果为(D )

A. a

B. b

C. c

D. cd

考察switch选择判断语句的使用

1.switch 语句选择的类型只有四种：byte，short，int ， char。JDK1.7添加了String数据类型

2.case 之间与 default 没有顺序。先判断所有的 case，没有匹配的 case 执行 default。

3.switch 语句停止的条件是遇到了 break 关键字或者结束 switch 语句的大括号。

4.如果匹配的 case 或者 default 没有对应的 break，那么程序会继续向下执行，运行可以执行的语句，直到遇到 break 或者 switch 结尾结束。

5.switch case 中的值必须要与 switch 表达式的值具有相同的数据类型。而且 case 后跟的值必须是常量，不能跟变量。case就像选择题的答案之一。 break就是如果该答案正确那么就可以跳出switch了,意思就是说 已经找出了正确的答案了。那么这道题也就做完了。如果 case没有匹配接着进行下一个 case 匹配,直到匹配为止。 最后如果都没有匹配上,那么 switch 给提供了一个默认的答案,就是 default。

注意:case后跟的是冒号:每个case中的执行语句一定要加 break；

* 以下代码输出是（D ）

|  |
| --- |
| int i=0;  int sum=0;  while(i<=10){  i++;  if( i%2 != 0 )  continue;  sum+=i;//2+4+6+8+10  }  System.out.println(sum); |

A. 55

B. 45

C. 35

D. 30

考察取模运算

5.%（取模，取余数） 取模运算的结果正负是取决于被除数的正负。

* 下列语句中执行跳转功能的语句是（C ）

A. for语句

B. while语句

C. continue语句

D. switch语句

考察循环语句中的continue关键字的作用

continue 的适用范围是循环语句中。

continue 关键字的作用： 跳过本次的循环体语句，继续下一次循环。

continue 关键字要注意的细节：

continue 关键字也可以配合标签使用。

continue 关键字后面也是不能直接跟语句的。

continue 关键字：语句将控制权传递给它所在的封闭迭代语句的下一次迭代。(跳出本循环，执行下一次循环)。

* 在switch(expression)语句中，expression的数据类型不能是（A ）

A. double

B. char

C. byte

D. short

考察switch选择判断语句的使用

1.switch 语句选择的类型只有四种：byte，short，int ， char。JDK1.7添加了String数据类型

2.case 之间与 default 没有顺序。先判断所有的 case，没有匹配的 case 执行 default。

3.switch 语句停止的条件是遇到了 break 关键字或者结束 switch 语句的大括号。

4.如果匹配的 case 或者 default 没有对应的 break，那么程序会继续向下执行，运行可以执行的语句，直到遇到 break 或者 switch 结尾结束。

5.switch case 中的值必须要与 switch 表达式的值具有相同的数据类型。而且 case 后跟的值必须是常量，不能跟变量。case就像选择题的答案之一。 break就是如果该答案正确那么就可以跳出switch了,意思就是说 已经找出了正确的答案了。那么这道题也就做完了。如果 case没有匹配接着进行下一个 case 匹配,直到匹配为止。 最后如果都没有匹配上,那么 switch 给提供了一个默认的答案,就是 default。

注意:case后跟的是冒号:每个case中的执行语句一定要加 break；

* 给出下列代码片段:

|  |
| --- |
| if ( x > 0 ) {  System.out.println(“Hello.”);  }else if ( x >-3 ) {  System.out.pirntln ( “ I am Tom. ”);  }else {  System.out.println (“How are you?”);  } |

请问将打印字符串 “How are you ?”的x的范围是（C ）

A. x>0

B. x > -3

C. x <= -3

D. x <=0 & x >-3

* 下列代码中，将引起编译错误的行是（ B）

|  |
| --- |
| 1）public class Exercise{  2） public static void main(String []args){  3） float f=0.0; //float值需要加f  4） f+=1.0;  5） }  6） } |

A. 第2行

B. 第3行

C. 第4行

D. 第6行

考察float数据类型的定义

注意: 一个小数如果没有加上任何的标识的时候，默认都double类型。如果是float类型的数据那么该数据必须要以f结尾。不区分大小写。

* 下列代码执行的结果为(A )

|  |
| --- |
| class Demo{  public static void main(String[] args){  int num = max(43,34);  System.out.println(num);  }  public static int max(int a,int b){  return a>b?a:b;  }  } |

A. 43

B. 23

C. 77

D. 9

考察函数的调用及函数的形参与实参的运算值传递运算

基本数据类型交换：发现交换值前后没有变量的值发生变化

原因：只是形参的值方式交换了

数组类型交换：交换值成功

操作的是同一个数组对象

对象的值交换：交换值成功

字符串的值交换:交换值失败

* 下面程序执行的结果是( A)

|  |
| --- |
| Class Demo{  public static void main(String [] args){  int a=10;  // 先判断，后自加  if(a++>10) {  a=20;  }  System.out.println(a);  }  } |

A. 11

B. 12

C. 20

D. 21

考察自增运算符

6．++(自增): 自增就是操作数+1

前自增: ++位于操作数的前面。 先自增，后使用。

后自增: ++位于操作数的后面。 先使用，后自增。

7．--（自减）:自减就是操作数-1

前自减：--位于操作数 的前面，先自减，后使用。

后自减: --位于操作数 的后面，先使用，后自减

如果运算符在变量的前面，则该变量自增 1 或者自减 1，然后返回的是变量的新值，如果运算符在变量的后面，则变量也会自增或者自减 1，但是返回的是变量原来的值。

++在前就是先运算，再取值，++在后就是先取值，再运算。

* 下面哪些是合法的标志符（BCD）

A. 2variable

B. variable2

C. what$

D. \_3\_

考察标识符的名命规则

标识符的要注意的细节

1.标识符的组成元素是字母（a-z A-Z）,数字（0~9） 、下划线（\_）、 美元符号（$）

2.标识符不能以数字开头。

3.java 的标示符是严格区分大小写的。

4.标识符的长度可以是任意。

5.自定义的标识符名称一般要有意义。

6.关键字、保留字(goto)不能用于自定义的标识符。

Java 中的标识符命名规范

1.包名: 多个单词组成时所有字母小写（例:package com.itcast）

2.类名和接口: 多个单词组成时所有单词的首字母大写(例:HelloWorld)

3.变量名和函数名: 多个单词组成时第一个单词首字母小写，其他单词首字母大写（例:lastAccessTime、getTime）。

4.常量名: 多个单词组成时，字母全部大写，多个单词之间使用\_分隔（例:INTEGER\_CACHE）

注意: 只是为了增加规范性、可读性而做的一种约定，标识符在定义的时候最好见名知意，提高代码阅读性

* 当整数变量值为多少时，只会输出" message 2" （BC）

|  |
| --- |
| switch(i){  case 1: System.out.println("message1");  case 2:  case 3: System.out.println("message2");  break;  } |

A. 1

B. 2

C. 3

D. 4

考察switch选择判断语句的使用

1.switch 语句选择的类型只有四种：byte，short，int ， char。JDK1.7添加了String数据类型

2.case 之间与 default 没有顺序。先判断所有的 case，没有匹配的 case 执行 default。

3.switch 语句停止的条件是遇到了 break 关键字或者结束 switch 语句的大括号。

4.如果匹配的 case 或者 default 没有对应的 break，那么程序会继续向下执行，运行可以执行的语句，直到遇到 break 或者 switch 结尾结束。

5.switch case 中的值必须要与 switch 表达式的值具有相同的数据类型。而且 case 后跟的值必须是常量，不能跟变量。case就像选择题的答案之一。 break就是如果该答案正确那么就可以跳出switch了,意思就是说 已经找出了正确的答案了。那么这道题也就做完了。如果 case没有匹配接着进行下一个 case 匹配,直到匹配为止。 最后如果都没有匹配上,那么 switch 给提供了一个默认的答案,就是 default。

注意:case后跟的是冒号:每个case中的执行语句一定要加 break；

* 下面哪些语句可以通过编译( ACD )

A. float a= 1.34f;

B. float b=1.0;

C. float c=2f;

D. float d=20f;

考察float数据类型的定义

注意: 一个小数如果没有加上任何的标识的时候，默认都double类型。如果是float类型的数据那么该数据必须要以f结尾。不区分大小写。

* 下列函数中能和函数int max(int a, int b, double c)构成重载关系的是(BC)

A. double max(int a, int b, double c)

B. void max(int a, double c, int b)

C. int max(double a, int b)

D. int max(int x, int y, double z)

考察函数重载的定义

函数重载的定义：在同一个类中，有一个以上的同名函数，只要函数的参数列表或参数类型不一样即可，与返回值无关， 这些统称为方法的重载。

函数的重载：函数的重载就是在一个类中出现了两个或者两个以上同名的函数。

函数的重载的作用： 一个函数名可以定义多个函数以应对各种不同类型的参数。

函数重载的要求：

1. 函数名必须一致。

2. 形参列表不一致（形式参数的个数不一致或者是对应的数据类型不一致）

3. 函数的重载与返回值类型无关及是否静态无关。

* 代码阅读，下列代码运行输出的结果为（AC）

|  |
| --- |
| int [] arr={23,54,76,87 };  for(int i=0;i<arr.length;i+=2){  System.out.println(arr[i]);  } |

A. 23

B. 54

C. 76

D. 87

* 下面数组定义正确的是？（D）

A. int arr[] = new int[3]; //Int不是关键字。

B. int arr[] = new int[3]{1,2,3}; //Int不是关键字。有元素参数就不可以有值。

C. int [][]x = new int[][]; //缺少数组维

D. int[][] x = new int[2][]; //创建两个NULL的空间，并且放在栈上。

* 下面的说法正确的是(AC )

A. byte ,short,char 数据类型可以自动转化为int

B. float 数据类型可以自动转换为long。

C. 在java语言中默认小数为double

D. byte a=23; byte b=12; a+b的结果为byte类型　//结果为int类型

考察表达式的数据类型自动提升

算术表达式，逻辑表达式

所有的 byte 型、short 型和 char 的值将被提升到 int 型。

如果一个操作数是 long 型，计算结果就是 long 型。

如果一个操作数是 float 型，计算结果就是 float 型。

如果一个操作数是 double 型，计算结果就是 double 型。

* 下面哪几段代码不会出现编译错误(AC)

A. boolean b=true;

boolean b2=true;

if(b==b2) { System.out.println("So true"); }

B. int i=0; if (i) { System.out.println("Hi"); }//if比较表达式结果必须为boolean

C. int i=1; int j=2; if(i==1|| j==2) System.out.println("OK");

D. int i=1; int j=2; if (i==1 &| j==2) System.out.println("OK");

考察逻辑运算符与if语句

if比较表达式结果必须为boolean

* 下面关于函数描述正确的是（AD）

A. 函数是对功能代码块的封装

B. 函数没有返回值的时候什么都不用写

C. 没有返回值的函数，不能有return 语句

D. 函数是可以没有形参的

考察函数的定义

1、定义函数可以将功能代码进行封装，函数的作用就是把一个功能代码封装起来，已达到功能代码可以被复用的效果。

2、便于对该功能进行复用

3、函数只有被调用才会被执行

4、函数的出现提高了代码的复用性

5、对于函数没有具体返回值的情况，返回值类型用关键字void表示，那么该函数中的return语句如果在最后一行可以省略不写。函数中只能调用函数，不可以在函数内部定义函数。

* 下面关于循环描述正确的是（AD）

A. while循环先判断循环条件，后执行循环操作

B. while 至少会执行一次

C. do-while先进行循环条件判断，后执行循环操作

D. do-while循环至少执行一次，后进行循环判断

do while 特点是条件无论是否满足，循环体至少被执行一次。

while:先判断条件，只有条件满足才执行循环体。

do while: 先执行循环体，再判断条件，条件满足，再继续执行循环体。

简单一句话：do while：无论条件是否满足，循环体至少执行一次。

注意一个细节 do while 后面的分号;

* Java的基本数据类型分为四型八种，分别指那些？我们选择数据类型的时候按照哪种原则选用？

整数类型：syte,shot、int、long

浮点数类型：float、double

布尔类型：boolean

字符类型：char

选择数据类型的时候尽量取占用字节少的，因为少的比较省内存空间。

* 编程实现时间转换: 5000秒是几小时几分钟几秒？请把代码粘贴到下面。

|  |
| --- |
| class Test{  public static void main(String[] args) {  int time=500;  int hour,minute,second;  hour=time/3600;  minute=(time-hour\*3600)/60;  second=time-hour\*3600-minute\*60;  System.out.println(time+"秒是:"+hour+"小时,"+minute+",:分钟,"+second+":秒");  }  } |

* 下列哪项不是JDK所包含的内容？（A）

A．Java编程语言

B．工具及工具的API

C．Java EE扩展API

D．Java平台虚拟机

* 下列关于JDK、JRE和JVM的描述。哪项正确？D

A．JDK中包含了JRE，JVM中包含了JRE

B．JRE中包含了JDK，JDK中包含了JVM

C．JRE中包含了JDK，JVM中包含了JRE

D．JDK中包含了JRE，JRE中包含了JVM

* 下列哪个工具可以编译java源文件？ A

A．javac

B．jdb

C．javadoc

D．junit

* JDK工具javadoc的作用是哪项？ A

A．生成Java文档

B．编译Java源文件

C．执行Java类文件

D．测试Java代码

* 以下哪些包是Java标准库中常用的包？（A,C,B）

A．java．lang

B．javax．servlet .http

C．j ava. io

D．java．sql

* 使用JDK工具生成的Java文档的文件格式是？ D

A．XML格式

B．自定义格式

c．二进制格式

D．HTML格式

* 以下关于JVM的叙述，哪项正确？（选两项） A C

A．JVM运行于操作系统之上，它依赖于操作系统

B．JVM运行于操作系统之上，它与操作系统无关

C．JVM支持Java程序运行，它能够直接运行Java字节码文件

D．JVM支持Java程序运行，它能够直接运行Java源代码文件

* 以下关于支持Java运行平台的叙述，哪项错误？ D

A．Java可在Solaris平台上运行

B．Java可在Windows平台上运行

C．Java语言与平台无关。Java程序的运行结果依赖于操作系统

D．Java语言与平台无关。Java程序的运行结果与操作系统无关

* 以下关于Applet和Java程序之间关系的叙述，哪项错误？ B

A． -个Applet就是一段Java程序

B．Applet是一种特殊的Java程序，它需要运行在Web服务器上

C．Applet是一种特殊的Java程序，它需要运行在Web浏览器上

D．Applet是一种Java SE平台的应用程序

* 以下关于Java HotSpot的描述，哪两项错误？（选两项） DA

A．JavaHotSpot是一种热编译技术，在编译Java源程序时会被使用

B．Java HotSpot是一种热编译技术，在运行Java代码时会被使用

C．Java HotSpot是一种热编译技术，它只对程序的部分字节码进行优化

D．Java HotSpot是一种热编译技术，它会对程序的全部字节码进行优化

* 环境变量PATH中含有多个路径时，路径和路径之间可以用哪项来隔开？A

A.： //用；分开

B. ，

C.\*

D:|

* CLASSPATH中的“．”的含义是哪项？B

A．省略号

B．当前目录

C．所有目录

D．上级目录

* JVM在执行一个Java类时，大致采用以下过程？ B

A．执行类中的代码一>装载类一>校验类

B．校验类一>装载类一>执行类中的代码

C．装载类一>执行类中的代码一>校验类

D．装载类一>校验类一>执行类中的代码

* 当运行—个Java程序时，传递参数的格式是哪项？ C

A． java类名参数1，参数2

B． javac类名参数1参数2

C． java类名参数1参数2

D． java类名参数1+参数2

* 以下关于Java文件名的叙述，正确的有？（选两项） A

A．Java源文件的扩展名应为.java

B．Java源文件的文件名应与文件中的类名一致

C． Java字节码文件的扩展名应为．java

D． 一个Java源文件中只能包含一个Java类

* 下列哪项不属于Java语言的基本数据类型？

A．int

B．String

C．double

D．boolean

* 下列哪项不是int类型的字面量？

A．\u03A6

B．077

C．OxABBC

D．20

* 下列哪项不是有效的标识符？ D

A．userName

B．2test

C．$change

D．\_password

* 下列哪项是Java语言中所规定的注释样式？（选三项） A C D

A．／／单行注释

B．--单行注释

C． ／\*

\*单行或多行注释

\*/

D． /\*\*

\*文档注释

\*/

* 下列哪项不是Java语言的关键字？

A．goto

B．sizeof

C．instanceof

D．volatile

* 现有如下五个声明：

Linel: int a\_really\_really\_really\_long\_variable\_name=5 ;

Line2: int \_hi=6；

Line3: int big=Integer. getlnteger("7”);

Line4：int $dollars=8;

line5: int %opercent=9;

哪行无法通过编译？ D

A．Line1

B．Line3

C. Line4

D. Line5

* 现有：

|  |
| --- |
| 1. class Top {  2． static int x=l；  3. public Top (int y) {  x\*=3;  }  4. }  5. class Middle extends Top {  6. public Middle() {  x+=1;  }  7. public static void main (String [] args) {  8. Middle m = new Middle();  9. System. out .println (x);  IO. }  II. } |

结果为：D

A.1

B. 2

C．3

D．编译失败

* 现有代码片段：

String s="123"；

String sl=S+456;

请问sl的结果是哪项？ A

A. 123456

B. 579

C．编译错误

D．运行时抛出异常

* 基本数据类型float的包裹类是哪项？ C

A． Integer

B． Double

C． Float

D． Character

* 现有：

|  |
| --- |
| 1． class Test4 {  2. public static void main (String [] args) {  3． boolean X = true;  4． boolean y = false;  5． short Z = 42;  7. if((z++ == 42) ＆＆ (y = true)){  z++;  }  8. if((x = false) || (++z == 45)){  z ++;  }  10. System. out.println(¨z=”+z);  II. }  12. } |

结果为： D

A. Z=42

B. z=44

C． Z= 45

D． z= 46

* 现有

|  |
| --- |
| class Test2 f  public static void main (String [] args) {  short a，b,c；  a=l;  b=2;  c=a+b；  a+=2：  }  } |

以上代码中，哪一句是错误的？ B

A．a=1：

B．C=a+b；

C． a+=2;

D. short a，b,C；

* 现有：

|  |
| --- |
| 5． class Wrench2 {  6． int size;  7. public static void main(String [] args) {  8. Wrench2 w=new Wrench2();  9． w.size=11;  IO. Wrench2 w2=go(w, w.size);  II. System. out .print (w2.size);  12． }  13. public static Wrench2 go(Wrench2 wr. int s) {  14. s=12;  15。 return wr;  16. }  17. } |

结果为： 11

A. 11

B. 12

c．编译失败。

D．运行时异常被抛出

* 现有：

|  |
| --- |
| 1． class Passer f  2． static final int x=55；  3. public static void main (String [] args) {  4． new Passer()．go (x);  5. System. out .print (x);  6， }  7． void go (int x) {  8． System. out .print(x++);  9． }  10. } |

结果是什么？ A

A．55

B．56

C．65

D．66

* 现有：

|  |
| --- |
| 1． class Wrench {  2. public static void main(String [] args) {  3. Wrench w=new Wrench();  Wrench w2=new Wrench();  4． w2=go (w, w2)；  5. System.out.print (w2==w);  6． }  7. static Wrench go (Wrench wrl, Wrench wr2) {  8. Wrench wr3=wrl;  wrl=wr2;  wr2=wr3;  9． return wr3;  10． }  11． } |

结果是什么？ B

A. false

B． true

C.编译失败

D．运行的时候有异常抛出

* 程序运行后的输出是哪项？D

|  |
| --- |
| class TestApp{  public static void main (String[] args){  for (int i=0; i<lO;i++){  if(i==3)  break;  System. out .print (i);  }  }  } |

A．0123

B．012456789

C． 0123456789

D． 012

* 上述程序运行后的结果是哪项？B

|  |
| --- |
| class TestApp{  public static void main (String[] args){  int x=6；  if (x>l)  System. out. println("x>l");  else if (x>5)  System. out .println("x>5");  else if (x<10)  System. out. println("x<lO");  else if (x<29)  System. out .println( "x<29");  else  System. out．println（“以上都不是”）;  }  } |

A．x>5

B．x>l

C． x<10

D．x<29

* 现有

|  |
| --- |
| class TestApp{  public static void main (String[] args){  int [5] myarray={ 10, 11, 12, 13, 14);  int sum=0;  for (int x ： myarray)  sum+=x;  System. out. println( "sum= "+sum);  }  } |

上述程序运行后的结果是哪项？C

A． sum=10

B． sum=70

C． sum=60

D． 运行时抛出异常

* 下列有关数组的声明中，正确的是哪项？（选两项） C D

A． int s[10];

B． int[10] s；

C． int[5] s={l,2,3,4,5};

D． int s[];

* 已知数组array，其最后一个元素的下标是？ A

A． array. size

B． array. length-l

C． array. size-l

D． array．length

* 程序

|  |
| --- |
| class TestApp{  public static void main (String[] args){  int X=5：  switch (x){  case l：  case 2：  case 3：  System. out．println（“一季度”）;  break;  case 4：  case 5：  case 6：  System. out．println（“二季度”）；  break;  default：  System. out．println（“三季度以上”）;  break;  )  }  ) |

上述程序运行后的结果是哪项？B

A. 一季度

B．二季度

c．三季度以上

D．无输出

* 为将数组myArray的长度由3改为6，现采取以下编码： D

int[]myArray=new int [3];

myArray=new int[6]；

代码执行后，以下叙述哪项是正确的？

A．数组myArray的长度已由3改为6，其中前3个元素的值不变，后3个元素

的值为空。

B．数组myArray的长度已由3改为6，其中前3个元素的值不变，后3个元素

需再经过初始化后才能使用。

C．数组myArray的长度没有变化。

D．数组myArray的长度已由3改为6，原来3个元素的值全部丢失。

* 现有：

|  |
| --- |
| 1． class Ifs {  2． public static void main (String [] args) {  3． boolean state=false;  4． int i=2；  5． if( (++i>2) ＆＆ (state=true))  6． i++;  7． if( (++i>4) l l (state=false))  8． i++;  9． System.out .println (i);  10． }  11． } |

结果为： A

A． 6

B． 5

C． 4

D．编译失败

* 现有：

|  |
| --- |
| 3． public class Tester {  4. public static void main (String[] args) {  5． int x = 5;  6. Integer xl = x; Integer x2 = x;  7. int x3=new Integer(5);  8. system.out.print(x1.equals(x));  9. system.out.print(xl==x);  lu. system.out.print(x2.equals(xl));  11. system.out.print(x2==xl);  12- system.out.print(x2==x3);  13- system.out.print(x2.equals(x3));  }  15．} |

结果为： C

A．编译失败

B. falsefalsetruetruetruetrue

C． truetruetruetruetruetrue

D. falsefalsetruetruetruefalse

E. truefalsetruefalsefalsetrue

F.运行时异常被抛出

* 现有：

|  |
| --- |
| 1． class Rectangle {  2' public static V.id main(string [] args) {  3． int[]x=(1，2，3);  4． x[1]= (x[1]>1) ?x[2]: O;  5． System.out．println(x[1]);  6． }  7． } |

结果为：A

A．3

B．2

C．1

D．0

* 现有：

|  |
| --- |
| 1．class Output (  2． public static void main (String[] args) {  3． int i=5：  4． System.out.print( "4"+i+"");  5． System.out.print (i+5+"7");  6． System.out.println (i+"8");  7． }  8． } |

结果为：C

A． 9 9722

B． 9 55758

C． 45 10758

D． 45 9722

* 以下哪种初始化数组的方式是错误的？

A． String[]names={"zhang", "wang", "li");

B． String names[] =new String [3];

names [O] ="zhang"; names [1] ="wang"; names [2] ="li";

C． String[3] names={"zhang", "wang", "li"};

D． 以上皆正确

* 现有：

|  |
| --- |
| 1． class WhileTests {  2． public static void main (String [] args) {  3． int X=5；  4． while (++x<4) {  5． --x;  6． }  7． System.out.println( "x="+x);  8． }  9． } |

结果是什么？ A

A.X=6

B. X=5

C.X=2

D．编译失败

* 现有：

|  |
| --- |
| 1． class Test2 {  2． public static void main (String [] args) {  3． boolean X= true;  4． boolean y= false;  5． short Z=20;  7． if((x==true) &＆ (y=true)) z++;  8． if((y==true) || (++z==22)) z++;  10． System. out .println( "z="+z)；  11． }  12． } |

结果是什么？C

A.Z=21

B. z=22

C. z=23

D． Z= 24

* 现有：

|  |
| --- |
| 1． class Foo {  2． public static void main (String [] args) {  3． int x = O；  4． int y = 4；  5． for (int z = 0; z < 3； Z++, X++) {  6． if( x > 1 & ++y < 10)  7． y++;  8． }  9． System. out .println (y);  10． }  11． } |

结果是什么？ A

A．7

B．8

C．10

D．12

* 配置PATH、CLASSPATH、JAVA\_HOME的作用？(10分)

保证javac命令可以在任意目录下运行。

保证class文件可以在任意目录下运行

方便修改内容是JDK的安装路径

* java的基本数据类型分为四型八种，分别指那些？我们选择数据类型的时候按照那种原则选用？(10分)

数据类型：byte、short、Int、long

浮点类型：float、double

布尔类型：boolean

字符类型：char

* 在JAVA中，下列哪些说法是正确的（ ） A C

A．java源文件的扩展名为.java

B．写好的java源程序可以直接运行

C．编写的源程序必须先编译后才能运行

D．程序员可以读懂扩展名为.class的文件

* 下列标识符哪个是合法的（ ） B

A．class

B．$abc

C．1234

D．\_name

* 下面哪些赋值语句是正确的?( ) D

A. long test = 012;

B. float f = -412;

C. int other = (int) true;

D. double d = 0x12345678;

E. byte b = 128;

* 在Java中，下列语句不能通过编译的有( ) B D

A. String s= “join”+ 3;

B. int a= “join”+3;

C. int a= ‘a’+5;

D. float f=5+5.5;

* 设 a = 3，则表达式 (--a )<<a的结果是（ ） B

A. 16

B. 8

C. 24

D. 12

* 下列代码运行的结果是( ) C

|  |
| --- |
| public static void main(String[] args) {  int x = 4;  int y = 5;  if(x++ > 4 & ++y > 5) {  x++;  }  System.out.println("x=" + x + ",y=" + y);  } |

A. x=4,y=5

B. x=4,y=6

C. x=5,y=6

D. x=6,y=6

* 尝试运行下面的代码会发生什么？( )

|  |
| --- |
| public class Test {  public static void main(String[] args) {  boolean m = true;  if (m = false) {  System.out.println("true");  } else {  System.out.println("false");  }  }  } |

A. 输出字符串”true”

B. 输出字符串”false”

C. 编译时错误

D. 运行时错误

* 编译并运行以下代码将发生什么？ （ ） E

|  |
| --- |
| public static void main(String args[]){  char digit = 'a';  for (int i = 0; i < 10; i++){  switch (digit)  {  case 'x' :  {  int j = 0;  System.out.println(j);  }  default :  {  int j = 100;  System.out.println(j);  }  }  }  int i = j;  System.out.println(i);  } |

A. 输出11次100

B. 输出10次100，然后抛出运行期异常

C. 编译错误，因为变量i不能在main() 方法中被声明2次

D. 编译错误，因为变量j不能在main() 方法中被声明2次

E. 以上都不对



|  |
| --- |
| class Demo B  {  public static int fun(int c)  {  return c+=2;  }  public static void main(String[] args)  {  int temp = fun(2);  System.out.println(temp);  }  } |

A. 2

B. 4

C. 6

D. 8

* 下面哪些函数是public void aMethod(){...}的重载函数?( ) B D

A. void aMethod( ){...}

B. public int aMethod(int a, float b){...}

C. public void aMethod (){...}

D. public float aMethod (int m){…}

* 在java中，关于数组描述正确的是（ ） B D

A．数组中的索引下标从1开始

B．存储在数组当中的数据都属于同一数据类型

C．通过数组名.length()能获得数组的长度

D．数组的最大索引下标是数组的长度减1

* 下面程序的输出结果是什么( ) D

int[] arr = new int[10];

System.out.println(arr[0]);

A．编译不通过

B．运行时出错

C．输出null

D．输出0

* 下面哪个语句正确地声明一个整型的二维数组( )

A. int a[][] = new int[][];

B. int b[10][10] = new int[][];

C. int c[][] = new int[10][10];

D. int []d[] = new int[10][10];

* 以下代码输出是（ ） D

|  |
| --- |
| class Demo {  public static void main(String[] args) {  int i = 0;  int sum = 0;  while (i <= 10) {  i++;  if (i % 2 != 0)  continue;  sum += i;  }  System.out.println(sum);  }  } |

A. 55

B. 45

C. 35

D. 30

* 和下面代码能完成相同的选项是（） C

|  |
| --- |
| class Demo {  public static void main(String[] args) {  int i = 1;  int sum = 0;  while (i <= 100) {  if (i % 2 == 0) {  sum = sum + i;  }  i++;  }  System.out.println(sum);  }  } |

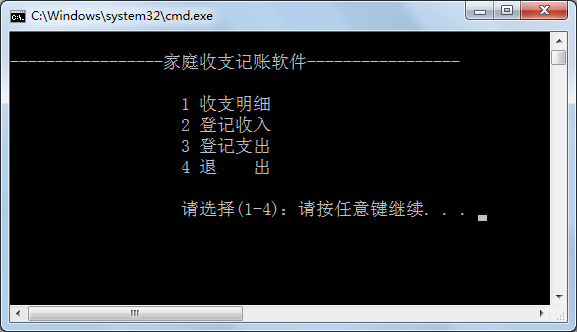
A. for (int x =1; x<=100;x++){ sum=sum+x;}

B. for (int x =0; x<=100;x+=2){ sum=sum+x;}

C. for (int x =1; x<=100;x+=2){ sum=sum+x;}

D.上述全对

* 项目一：家庭收支记账软件



|  |
| --- |
| import java.util.Scanner;  /\*\*  Utility工具类：  将不同的功能封装为方法，就是可以直接通过调用方法使用它的功能，而无需考虑具体的功能实现细节。  \*/  public class Utility {  private static Scanner scanner = new Scanner(System.in);  /\*\*  用于界面菜单的选择。该方法读取键盘，如果用户键入’1’-’4’中的任意字符，则方法返回。返回值为用户键入字符。  \*/  public static char readMenuSelection() {  char c;  for (; ; ) {  String str = readKeyBoard(1);  c = str.charAt(0);  if (c != '1' && c != '2' && c != '3' && c != '4') {  System.out.print("选择错误，请重新输入：");  } else break;  }  return c;  }  /\*\*  用于收入和支出金额的输入。该方法从键盘读取一个不超过4位长度的整数，并将其作为方法的返回值。  \*/  public static int readNumber() {  int n;  for (; ; ) {  String str = readKeyBoard(4);  try {  n = Integer.parseInt(str);  break;  } catch (NumberFormatException e) {  System.out.print("数字输入错误，请重新输入：");  }  }  return n;  }  /\*\*  用于收入和支出说明的输入。该方法从键盘读取一个不超过8位长度的字符串，并将其作为方法的返回值。  \*/  public static String readString() {  String str = readKeyBoard(8);  return str;  }    /\*\*  用于确认选择的输入。该方法从键盘读取‘Y’或’N’，并将其作为方法的返回值。  \*/  public static char readConfirmSelection() {  char c;  for (; ; ) {  String str = readKeyBoard(1).toUpperCase();  c = str.charAt(0);  if (c == 'Y' || c == 'N') {  break;  } else {  System.out.print("选择错误，请重新输入：");  }  }  return c;  }  private static String readKeyBoard(int limit) {  String line = "";  while (scanner.hasNext()) {  line = scanner.nextLine();  if (line.length() < 1 || line.length() > limit) {  System.out.print("输入长度（不大于" + limit + "）错误，请重新输入：");  continue;  }  break;  }  return line;  }  } |
| class FamilyAccount{  public static void main(String[] args){  boolean isFlag = true;  //用于记录用户的收入和支出的详情  String details = "收支\t账户金额\t收支金额\t说 明\n";  //初始金额  int balance = 10000;  while(isFlag){  System.out.println("-----------------家庭收支记账软件-----------------\n");  System.out.println(" 1 收支明细");  System.out.println(" 2 登记收入");  System.out.println(" 3 登记支出");  System.out.println(" 4 退 出\n");  System.out.print(" 请选择(1-4)：");  //获取用户的选择：1-4  char selection = Utility.readMenuSelection();  switch(selection){    case '1':  //System.out.println("1.收支明细");  System.out.println("-----------------当前收支明细记录-----------------");  System.out.println(details);  System.out.println("--------------------------------------------------");  break;  case '2':  //System.out.println("2.登记收入");  System.out.print("本次收入金额：");  int addMoney = Utility.readNumber();  System.out.print("本次收入说明：");  String addInfo = Utility.readString();    //处理balance  balance += addMoney;  //处理details  details += ("收入\t" + balance + "\t\t" + addMoney + "\t\t" + addInfo + "\n");  System.out.println("---------------------登记完成---------------------\n");  break;  case '3':  //System.out.println("3.登记支出");    System.out.print("本次支出金额：");  int minusMoney = Utility.readNumber();  System.out.print("本次支出说明：");  String minusInfo = Utility.readString();    //处理balance  if(balance >= minusMoney){  balance -= minusMoney;    //处理details  details += ("支出\t" + balance + "\t\t" + minusMoney + "\t\t" + minusInfo + "\n");  }else{  System.out.println("支出超出账户额度，支付失败");  }  System.out.println("---------------------登记完成---------------------\n");  break;  case '4':  //System.out.println("4.退 出");  System.out.print("确认是否退出(Y/N)：");  char isExit = Utility.readConfirmSelection();  if(isExit == 'Y'){  isFlag = false;  }  //break;  }  }  }  } |