# 运算符

(布尔类型)?值1:值2 三元运算符

值1和值2需要类型一致

例 ：

int a = (false)?10:5;

# 获取用户的输入

要使用Scanner类

1. import java.util.Scanner;
2. Scanner s = new Scanner(System.in);
3. s.nextInt() 获取整数
4. s.next() 获取字符串

# 流程控制

顺序流程： 按照代码的顺序来执行

## 条件控制

语法：

if (布尔条件) {

代码

} else {

如果if 的条件不成立，会执行else块内的代码

}

if(条件1）｛

代码块1

｝else　if(条件2) {

代码块2

} else if(条件3) {

代码块3

} else if(条件n) {

} else {

else 的代码块

}

当条件1成立时，执行代码块1

当条件1不成立，当条件2成立时，执行代码块2

当条件1，2不成立，条件3成立时，执行代码块3

...

当条件1...n都不成立，执行else 的代码块

if (条件) 当代码只有一行时

switch(变量) {

case 值1: 语句1 break;

case 值2: 语句2 break;

case 值3: 语句3 break;

case 值n: 语句n break;

default: 语句d

}

注意：用来执行switch匹配的变量只能是 byte,char,short, int, String(jdk 7.0支持)

case 后的匹配只能是一个值，不能是变量

## 循环控制

重复执行相同的或相似的代码

## while （重点）

语法：

while(条件) {

重复执行的代码

}

// 条件为真，执行while语句块中的代码

// 直到条件为假，退出while循环

## do... while

do {

要重复执行的代码

} while (条件)

// 条件为真，执行语句块中的代码

// 直到条件为假，退出循环

与while的区别：

do...while它是先执行语句块内容，再判断条件 (至少会执行一次)

while 是先判断条件，条件成立，才执行循环语句块内容

## for (重点)

语法：

for(赋值表达式 ; 条件判断部分; 每次循环结束时执行部分) {

重复执行的代码

}

## break

退出整个循环

## continue

跳过本次循环

a: for() {

b: for() {

countinue 编号;

}

}

#### 循环嵌套

//3 \* 3 = 9

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

System.out.println("hello")

}

}

练习题：

1. 有三个int变量，求出它们的最大值

|  |
| --- |
| public static void main(String[] args) {  // 有三个int变量，求出它们的最大值  Scanner s = new Scanner(System.in);    // 读入三个变量的值  int a = s.nextInt(); // 1  int b = s.nextInt(); // 3  int c = s.nextInt(); // 4    // 进行比较  int max = a; // 1  if(max < b) {  max = b;  }  if(max < c) {  max = c;  }  System.out.println("最大值是：" + max);  } |

1. 当货物重量小于20 公斤的时候，收费5 元，大于20 公斤小于100 公斤的时 候超出20 公斤的部分按每0.2 元每公斤计费，如果超出100 公斤的时候，超出的部分按照每公斤0.15 元计算。 读入货物的重量，输出计算之后货物的运费

|  |
| --- |
| public static void main(String[] args) {  Scanner s = new Scanner(System.in);  // 读入用户输入的重量  int weight = s.nextInt();  double fee = 0.0;    if(weight <= 20 ) {  fee = 5.0;  } else if (weight > 20 && weight <= 100) {  fee = 5.0 + (weight-20) \* 0.2;  } else if (weight > 100) {  fee = 5.0 + 80 \* 0.2 + (weight-100) \* 0.15;  }  System.out.println("运费是:" + fee);  } |

1. 三个数字，按由大到小的顺序输出
2. 计算1+3+5+...+99的和

|  |
| --- |
| public static void main(String[] args) {  // for(int i=1; i<=99; i=i+2)  int sum = 0;  for(int i=1; i<=99; i+=2) {  System.out.println(i);  sum = sum + i;  }    System.out.println("结果是:" + sum);  } |

1. 计算一个小于10的正整数的阶乘

|  |
| --- |
| // 计算10以内的阶乘  public static void main(String[] args) {  Scanner s = new Scanner(System.in);  int n = s.nextInt(); //6  int result = 1;  for(int i = 1; i <= n; i++) {  result = result \* i;  }  System.out.println(n + "的阶乘是:" + result);  // int 最大存储13的阶乘    }  Public static void main(String[] args){  Scanner s=new Scanner(System.in);  Int n=s.nextInt();  count(n);  }  Public static int count(int n){  If(n=1){  return 1;  }else{  Return n\*n-1;  }  } |
|  |

6\* 5 \*4 \*3 \*2 \*1

1. 搬砖问题：36 块砖，36 人搬，男搬4，女搬3，两个小孩抬1 砖， 要求一次全搬完，问男、女和小孩各若干？

穷举法

类似题目：

“百钱买百鸡”是我国古代的著名数学题。题目这样描述：

3 文 钱可以买1只公鸡，

2 文钱可以买一只母鸡，

1 文钱可以买3 只小鸡。用100 文 钱买100 只鸡，那么各有公鸡、母鸡、小鸡多少只

|  |
| --- |
| **public** **static** **void** main(String[] args){  **for**(**int** i=0;i<9;i++){  **for**(**int** j=0;j<12;j++){  **int** k=36-i-j;  **if**((4\*i+3\*j+k/2)==36&&(36-i-j)%2==0)System.*out*.println(i+","+j+","+(36-i-j));  }  }  } |

1. 读入一个整数n，输出如下图形

n = 3

\*

\*\*\*

\*\*\*\*\*

n = 4

\*

\*\*\*

\*\*\*\*\*

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

|  |
| --- |
| **public** **static** **void** huihua(**int** n){  **for**(**int** i=1;i<=n;i++){  **for**(**int** j=1;j<=(n-i);j++){  System.*out*.print(" ");  }  **for**(**int** k=1;k<=(2\*i-1);k++){  System.*out*.print("\*");  }  System.*out*.println();  }  } |

1. 输出一个九九乘法表

|  |
| --- |
| **public** **static** **void** huihua(**int** n){  **for**(**int** i=1;i<n;i++){  **for**(**int** j=1;j<=i;j++){  System.*out*.print(i+"\*"+j+"="+i\*j+" ");  }  System.*out*.println();  }  } |

1. 中 国 古 代 数 学 家 研 究 出 了 计 算 圆 周 率 最 简 单 的 办 法:PI=4/1-4/3+4/5-4/7+4/9-4/11+4/13-4/15+4/17......这个算式的结果会无 限接近于圆周率的值,我国古代数学家祖冲之计算出,圆周率在3.1415926 和 3.1415927 之间,请编程计算,要想得到这样的结果,他要经过多少次加减法运算?

|  |
| --- |
| public static void main(String[] args) {  double pi = 0.0;  double i = 1;  long start = System.currentTimeMillis(); // 当前时间, 毫秒值  // pi > 3.1415926 && pi < 3.1415927  while(pi <= 3.1415926 || pi >= 3.1415927) {  double a = 4.0; //分子  double b = 2.0\*i-1.0; // 分母  if(i % 2 == 0) {  // i是偶数 \* -1  pi = pi - a / b;  } else {  pi = pi + a / b;  }  i++;  }  long end = System.currentTimeMillis();  System.out.println("共花费:" + (end-start));  // i 是奇数\*1 , i 是偶数 \*-1  System.out.println(pi);  System.out.println("循环了" + i + "次");      // if(a > 5 && a < 7)  // a <=5 || a >= 7  } |

1. 已知：faibonacci（费波那契）数列的前几个数分别为0，1，1，2，3，5，……。从第3 项开始，每一项都等于前两项的和。 读入一个整数n，编程求出此数列的前n 项。 注意：假设前两个数0,1已知

|  |
| --- |
| public static void main(String[] args) {  int n = 8;  // n=6 0，1，1，2，3，5  // n=7 0，1，1，2，3，5，, 8  int n\_2 = 0;  int n\_1 = 1;  System.out.print("0 1");  for(int i=3; i<=n; i++) {  int current = n\_2 + n\_1;  System.out.print(" "+current);  n\_2 = n\_1;  n\_1 = current;  }  } |