* 开发环境

1.Linux:

开源的操作系统、免费、服务器

目录、挂载点、高

pwd,ls,cd

绝对:根 相对:当前目录 .(当前目录) ..(上一级目录)

2.java开发环境:

1)编译运行过程:

1.1)编译期:.java，经过编译，生成.class字节码文件

1.2)运行期:JVM加载.class并运行.class

跨平台，一次编程到处使用

2)名词:

2.1)JVM:java虚拟机

加载.class并运行.class

2.2)JRE:java运行环境

除了包含JVM以外还包含运行java程序所必须的环境

JRE=JVM+java系统类库

2.3)JDK:java开发工具包

除了包含JRE以外还包含开发java程序所必须的命令工具

JDK=JRE+编译、运行等工具

运行的最小环境为JRE，开发的最小环境为JDK

3)配置环境变量:

3.eclipse:

1)新建java项目

2)新建java包

3)新建java类，main()中:

System.out.println("Hello");

* 变量、基本数据类型

1.变量:存数的，代词，指代的就是它所存的那个数

1)声明:

int a; //声明一个整型的变量，名为a

int b,c,d; //声明三个整型的变量，名为b,c,d

2)命名:

2.1)只能包含字母、数字、\_和$符，并且不能以数字开头

2.2)严格区分大小写

2.3)不能使用关键字

2.4)可以中文命名，但不建议

建议:"见名知意"、"驼峰命名法"

3)初始化:第一次赋值

3.1)声明的同时初始化

int a = 5; //声明整型变量a并赋值为5

3.2)先声明后初始化

int a; //声明整型变量a

a = 5; //给变量a赋值为5

4)使用:

4.1)对变量的使用就是对它所存的那个数的使用

4.2)变量的使用要与数据类型匹配

int a = 3.14; //编译错误，数据类型不匹配

4.3)变量在用之前必须声明并初始化

System.out.println(m); //编译错误，m未声明

int m;

System.out.println(m); //编译错误，m未初始化

2.基本数据类型:

1)int:整型，只能装整数，4个字节，-21个多亿到21个多亿

1.1)整数直接量默认为int型，但不能超范围，超范围则编译错误

1.2)两个整数相除，结果还是整数，小数位无条件舍弃

1.3)整数运算时超出范围则发生溢出，溢出是需要避免的

2)long:长整型，只能装整数，8个字节，很大很大很大

2.1)长整型直接量需要数字后加L或l

2.2)运算时若有可能溢出，建议在第1个数字后加L，不溢出可以不加L

2.3)System.currentTimeMillis()用于获取自

1970.1.1零时到此时此刻的毫秒数

3)double:浮点型，只能装小数，8个字节，很大很大很大

3.1)浮点数直接量默认为double型，若想表示float需要数字后加F或f

3.2)double型数据参与运算时，会出现舍入误差，所以精确运算场合不能用

4)boolean:布尔型，1个字节

4.1)只能取值为true和false

5)char:字符型，2个字节

5.1)java中的char采用Unicode编码格式，

一个字符char对应一个码int

表现的形式是char字符，但实质上存储的是int码

码只能在0到65535之间

ASCII码:'a'--97 'A'--65 '0'--48

5.2)字符直接量必须放在单引号中，有且仅有一个

5.3)特殊字符需要通过\来转义

3.基本数据类型间的转换:

基本数据类型从小到大依次为:

byte,short,int,long,float,double,char,

1)两种方式:

1.1)自动类型转换:从小类型到大类型

1.2)强制类型转换:从大类型到小类型

语法: (要转换为的类型)变量

强转有可能会溢出或精度丢失

2)两点规则:

2.1)整数直接量可以直接赋值给byte,short,char，

但不能超范围

2.2)byte,short,char型数据参与运算时，

先一律转换为int再运算

4.Scanner的用法:

1)在package下:

import java.util.Scanner;

2)在main()方法中:

Scanner scan = new Scanner(System.in);

3)在第2步之下:

System.out.println("请输入年龄:");

int age = scan.nextInt();

System.out.println("请输入单价:");

double price = scan.nextDouble();

* 运算

1.运算符:

1)算术:+,-,\*,/,%,++,--

2)关系:>,<,>=,<=,==,!= boolean

3)逻辑:&&,||,! boolean

4)赋值:=,+=,-=,\*=,/=,%=

5)字符串连接:+

6)条件(三目)运算:boolean?数1:数2

2.分支结构:

1)if结构:1条路

2)if...else结构:2条路

3)if...else if结构:多条路

4)switch...case结构:多条路

优点:效率高、结构清晰

缺点:整数、相等

case中如果没有break则一直走到完

5）break:退出switch

用switch能实现的，用if...else if都可以实现

用if...else if能实现的，不一定能用switch实现

3.闰年的判断公式:

1)能被4整除，并且，不能被100整除，或者能被400整除

(year%4==0 && year%100!=0) || year%400==0

2）输出系统当前时间

long startTime = System.currentTimeMillis()；

4.循环:

1.循环:反复执行一段相同或相似的代码

2.循环三要素:

1)循环变量的初始化

2)循环的条件(以循环变量为基础)

3)循环变量的改变(向着循环的结束变)

循环变量:在循环过程中所改变的那个数

3.循环结构:

1)while:先判断后执行，有可能一次都不执行

2)do...while:先执行后判断，至少执行一次

第1要素与第3要素相同时首选

3)for:应用率最高，固定次数循环

4.break:退出循环

continue:跳过循环体中剩余语句而进入下一次循环

5.三种循环的更佳适用情况:

1)while:"当..."循环

2)do...while:"直到..."循环

第1要素与第3要素相同时首选do...while

3)for:固定次数循环，应用率最高

6.嵌套循环:

1)循环中套循环，一般多行多列时使用，

外层控制行，内层控制列

2)执行规则:外层循环走一次，内层循环走所有次

3)建议:嵌套的层数越少越好，能用一层就不用两层

若必须超过三层才能实现，说明设计有问题

4)break只能跳出一层循环

7.程序=算法+数据结构

1)算法:解决问题的流程/步骤(顺序、分支、循环)

2)数据结构:将数据按照某种特定的结构来保存数怎么存

设计良好的数据结构会导致好的算法

8.任何复杂的程序逻辑都可以通过三种结构来实现:

1)顺序结构:从上往下，逐行执行，每句必走

2)分支结构:有条件的执行某语句一次，并非每句必走

3)循环结构:有条件的执行某语句多次，并非每句必走

* 数组

1)相同数据类型元素的集合

2)是一种数据类型(引用类型)

3)数组的定义:

int[] arr = new int[4];

4)数组的初始化:

int[] arr = new int[4]; //0,0,0,0

int[] arr = {1,4,5,8}; //1,4,5,8

int[] arr = new int[]{1,4,5,8}; //1,4,5,8

int[] arr;

arr = {1,4,5,8}; //编译错误

arr = new int[]{1,4,5,8}; //正确

5)数组的访问:

5.1)通过(数组名.length)可以获取数组的长度

int[] arr = new int[5];

System.out.println(arr.length); //5

5.2)通过下标来访问数组中的元素

下标从0开始，最大到(数组.length-1)

int[] arr = new int[3];

arr[0] = 100; //给arr中第1个元素赋值为100

arr[1] = 200; //给arr中第2个元素赋值为200

arr[2] = 300; //给arr中第3个元素赋值为300

arr[3] = 400; //数组下标越界异常

System.out.println(arr[arr.length-1]); //输出arr中的最后一个元素

6)数组的遍历:

（1）直接输出方法

System.out.println( Arrays.toString(arr))

（2）for循环输出方法

int[] arr = new int[10];

for(int i=0;i<arr.length;i++){

arr[i] = (int)(Math.random()\*100);

}

for(int i=0;i<arr.length;i++){ //正序

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

}

for(int i=arr.length-1;i>=0;i--){ //倒序

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

}

7)数组的复制:

7.1)System.arraycopy(a,1,a1,0,4); //把a数组里的从第2个数起的4个数，复制到a1里面，从第1个开始

7.2)int[] a1 = Arrays.copyOf(a,6);

a = Arrays.copyOf(a,a.length+1); //扩容

8)数组的排序:

8.1)Arrays.sort(arr); //升序，效率高

8.2)冒泡排序:

8.2.1)四个数冒三轮

8.2.2)每一轮都是从第1个元素开始冒

每一次都是和它的下一个元素比

8.2.3)冒出来的就不带它玩了

for(int i=0;i<arr.length-1;i++){ //控制轮

for(int j=0;j<arr.length-1-i;j++){ //控制次

if(arr[j]>arr[j+1]){ //和它下一个元素比

int t=arr[j]; //若满足则交换

arr[j]=arr[j+1];

arr[j+1]=t;

}

* 方法:函数、过程

1.方法:函数、过程

1)用于封装一段特定的业务逻辑功能

2)方法应尽可能的独立，一个方法只干一件事

3)方法可以被反复调用多次

4)减少代码重复，便于程序的维护，便于团队的协作开发

2.方法的定义:

修饰词 返回值类型 方法名(参数列表){

方法体

}

3.方法的调用:

1)无返回值: 方法名(有参传参);

2)有返回值: 数据类型 变量 = 方法名(有参传参);

后期还需要用到方法中的某个数据----有返回值

后期不再需要用到方法中的某个数据--无返回值

4.return:

1)return 值; //1.1)结束方法的执行 1.2)返回结果给调用方

2)return; //2.1)结束方法的执行

5.其他

double c = Math.random();获取随机数

double d = Math.sqrt(25);//求（25）的平方跟