关键字

分号：在Java中 ,有一些字符被当作分隔符使用,最常用的分隔符是分号();，用来分隔语句

|  |  |
| --- | --- |
|  | **关键字** |
| 类  | enum、interface、class、extends、implements  |
| 对象  | new、instanceof、this、super |
| 包  | package、import |
| 数据类型  | byte、short、int、long、float、double、char、boolean |
| 分支  | if、 else、switch、case、break、continue |
| 循环  | do、while、for |
| 方法  | void、return |
| 异常  | throw、throws、try、catch、finally |
| 修饰符  | Abstract、final、private、protected、public、、static、synchronized、strictfp、native、assert、transient、volatile |
| 保留字  | const、goto  |

关键字都是小写字母

关键字不要作为标识符使用

空白：在代码中遇到运算符时，例如 = 等，两端留出空格，增强代码可读性。

代码行之间，可以根据逻辑关系加空白行，增强可读性

{ }花括号

* + 用来包括自动初始化的数组的值
	+ 用来定义程序块,类,方法以及局部范围

Java注释

* + // 单行注释
	+ /\* 单、多行注释 \*/
	+ /\*\* 文档注释 \*/

**2 标识符的概念**

**给变量命名是尽量做到见名知义。**

* + Java语言中，对于变量，常量，方法，都有名字，统统称之为Java标识符。
	+ 标识符是用来给类、对象、方法、变量、接口和自定义数据类型命名的。

标识符的命名规则：

* + 某一个区域中是唯一的，在不同的区域中可以使用同一名字
	+ 必须由字母、数字、下划线和$符号组成
	+ 不能以数字开头
	+ 不能使用非法的字符，如：#，%.......&等
	+ 不能使用系统关键字
	+ 不能使用空格来分隔
	+ 长度无限制
	+ 严格区分大小写
	+ 不同的标识符有一定的命名规矩
	+ 对于常量的命名要注意初始值

例如：

myName,$points, \_sys\_ta是合法的

#name，25name，class，&time

是非法的

**标识符的命名约定**

* + **类和接口名：每个字的首字母大写,例如MyClass，HelloWorld等。**
	+ **方法名。首字母小写，其余的首字母大写，尽量少用下划线。例如，myName，setTime等。这种命名方法叫做驼峰式命名。**
	+ **常量名。基本数据类型的常量名使用全部大写字母，字与字之间用下划线分隔。对象常量可大小混写。例如，SIZE\_NAME。**
	+ **变量名。可大小写混写，首字母小写，字间分隔符用字的首字母大写。不用下划线，少用美元符号。**

**3 包（package）**

包的概念：

* + 物理上是文件夹
	+ 逻辑上是有逻辑关系的类的集合

包的作用：

* + 避免类重名
	+ 控制访问权限

包（package）的命名规范：

在包名中，可以使用.号来区分包的级别；包名一般情况下是小写

第一级 指该项目的类型，如com,org,gov等，

第二级 指项目所开发或者运行的公司名称，如：chinasofti,icss,huawei等

第三级 指项目的名称，如：corejava,bcms,oa,erp等

第四级 指项目模块的名称，如：bean,action,exception,chapter01等

编译Trainer后，其类文件将被编译在com/chinasofti/corejava/chapter03目录下

package com.chinasofti.corejava.chapter03;

 public class Trainer{

 }

**包的作用**

包可以更好得管理逻辑相关的类，可以控制不同包之间的访问权限

导入外部包的类，关键字“import”

 import passageName[1].passageName[2]….className;

package com.chinasofti.corejava.chapter03;

 Import com.chinasofti.corejava.chapter04.Trainer;

 public class Test{

 public static void main(String[] args){

 }

 }

**4 访问权限**

1. 类（专指外部类）只可以是同包的default，或者公共的public

 属性、方法可以选择四种权限的任意一个。但是实际中，属性多用private，方法、构造方法多用public

Java语言有四个权限访问修饰符，权限从大到小依次为：

1）public ：公共权限 修饰类、属性、方法。可以被任意类访问

2）protected：受保护的权限 修饰属性、方法。

 可以被同包类访问，如果不是同包类，必须是该类的子类才可以访问。

3）default：同包权限 修饰类、属性、方法。只能被同包的类访问

4）private：私有权限 修饰属性、方法。 只能在本类中访问

自行编写Java类，在该类中定义四个方法，访问权限分别是public,protected,(default),private,在另一个类中分别访问这四个方法。两个类在不同包里。查看编译情况。

package com.chinasofti.a;

public class A {
 public void f1(){
 }
 protected void f2(){
 }
 void f3(){
 }
 private void f4(){
 }
}

package com.chinasofti.b;

import com.chinasofti.a.A;

public class B {
 public static void main(String[] args){
 A a=new A();
 a.f1();
 a.f2();
 a.f3();
 a.f4();
 }
}

自行编写Java类，体现封装性的含义。

public class Person {
 private String name;
 private int age;

 public void setAge(int age){
 if(age>=0){
 this.age=age;
 }
 }
}

Java中的流程

* + 顺序流程
	+ 分支流程
	+ 循环流程

条件语句 — if

* + 语法格式 — if ( 单一条件)



条件语句 — if （说明）

* + 条件表达式可以是任何一种逻辑表达式
	+ 如果表达式值为true，则执行花括号的内容后，再执行后面的语句
	+ 如果表达的值为false，则直接执行后面的语句
	+ 如果，大括号只有一条语句，则可以省略大括号（推荐不省略）

条件语句 — if…else…

* + 语法格式 — if …else… ( 二选一)

变量 = 布尔表达式 ? 语句1: 语句2

ü 如果布尔表达式的值为true，则执行语句1

ü 如果布尔表达式的值为false，则执行语句2



条件语句 — if…else…

* + 示例

int x = 2;

if(x<2){

x++;

}

else{

x--;

}

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

int x = 2;

x=x<2?++x:--x;

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

条件语句 — if…else if…

* + 语法格式 — if …else if…( 多选一)，可嵌套多个else if



条件语句 — if…else if…

* + 示例

int x = 88;

if(x<60){

System.out.println(“成绩不合格”);

}

else if(x>=85){

System.out.println(“成绩优秀”);

}

else{

System.out.println(“成级良好”);

}

条件语句 — switch

* + 语法格式 — switch

switch (表达式)

{

case 取值1：语句块1

…

case 取值n：语句块n

default: 语句块n+1

}

条件语句 — switch（说明）

* + 表达式的值只可以接受int、byte、char、short 型、枚举，JDK1.6以后String也可以，不接受其他类型的值，不允许有重复的case取值
	+ switch一旦碰到第一次case匹配，程序就会跳转到这个标签位置
	+ 开始顺序执行以后所有的程序代码，而不管后面的case条件是否匹配，直到碰到break语句为止

条件语句 — switch

* + 示例

int x = 2;

switch (x) {

case 1:

System.out.println("Monday");

break;

case 2:

System.out.println("Tuesday");

break;

case 3:

System.out.println("Wednesday");

break;

default:

System.out.println("Sorry,I don't know");

}

循环语句 — while

while (条件表达式)

{

执行语句块

}



循环语句 — while

* + 示例

 需求：计算1+2+3+…+10的和（提示利用while语句）

**int x=1;**

**int sum = 0;**

**while(x<11){**

 sum = sum + x;

 x++;

}

System.***out.println(sum);***

循环语句 — do…while 至少会执行一次循环

* + 语法格式

do

{

执行语句块；

} while(条件表达式 );



循环语句 — do…while

int x=3;

do{

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

x++;

} while(x<3);

循环语句 — for

for(初始化表达式；循环条件表达式；循环后的操作表达式)

{

执行语句块

}



循环语句 — 死循环

for(int x=1;;x++)

{

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

}

循环的中断语句 — break 、continue

* + 中断语句
	+ 多重循环中可以使用标号，指定中断的具体循环

在使用循环语句时，只有循环条件表达式的值为false时，才能结束循环。有时，我们想提前中断循环，要实现这一点，只需要在循环语句块中添加break或continue语句

中断语句 — break

* + break语句用于终止某个语句块的执行。用在循环语句体中，可以强行退出循环。
	+ “break;”语句：可以出现在while、do…while、for、switch语句体中。
	+ “break label”语句 ：可以出现在任何语句体中。多数还是使用在循环中。

中断语句 — break

* + 示例1

 需求：累加求和：1+2+3+……直到和为666结束循环

int i , sum;

for(i=1;i<101;i++){

sum+=i;

if(sum>=666){

break;

 }

}

System.out.println(“从1到”+i+“的和为sum”);

中断语句 — break

public class Test {

public static void main(String[] args) {

outer: for (int i = 0; i < 10; i++) {

System.out.println("Outer loop");

inner: while (true) {

Scanner sc = new Scanner(System.in);

String s = sc.next();

System.out.println("inner Loop:" + s);

if (s.equals("hello"))

break inner;

if (s.equals("kitty"))

break outer;}

}

}

}

中断语句 — continue

* + continue语句用在循环语句体中，用于终止某次循环过程，跳过循环体中continue语句下未执行的循环，开始下一次循环过程
	+ “continue;” 语句：只能出现在循环语句while、do…while、for中

for(int i=1;i<50;i++) {

if((i%2)==0){

continue;

}

System.out.print(i+" ");

}