1. 练习if else用法，如果去游泳，我就穿泳衣，如果去打排球，我就穿运动服，否则我穿牛仔裤；

**package** com.test4;

**import** java.util.Scanner;

**public** **class** Home1 {

**public** **static** **void** main(String[] args) {

String a="游泳";

String b="打排球";

System.***out***.println("明天穿什么");

Scanner in=**new** Scanner(System.***in***);

String c=in.next();

**if**(c.equals(a)){

System.***out***.println("穿泳衣");

}

**else** **if**(c.equals(b)){

System.***out***.println("穿运动服");

}

**else**{

System.***out***.println("穿牛仔裤");

}

}

}

1. 练习switch case 用法，键盘输入字符串，如果是A 打印优秀，如果是B，打印良好，如果是C，打印及格，否则打印不及格；

**package** com.test4;

**import** java.util.Scanner;

**public** **class** Home2 {

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

String a = in.next();

**switch** (a) {

**case** "A":

System.***out***.println("优秀");

**break**;

**case** "B":

System.***out***.println("良好");

**break**;

**case** "C":

System.***out***.println("及格");

**break**;

**default**:

System.***out***.println("不及格");

**break**;

}

}

}

1. 练习for，while， do while用法：
2. 分别用这三个控制语句打印100以内能被3或者5整除的数；

For：

**package** com.test4;

**public** **class** Home3 {

**public** **static** **void** main(String[] args) {

**for** (**int** a = 1; a <= 100; a++) {

**if** (a % 3 == 0 || a % 5 == 0) {

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

}

}

}

}

While：

**package** com.test4;

**public** **class** Home3 {

**public** **static** **void** main(String[] args) {

**int** a = 1;

**while** (a <= 100) {

**if** (a % 3 == 0 || a % 5 == 0) {

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

}

a++;

}

}

}

do while:

**package** com.test4;

**public** **class** Home3 {

**public** **static** **void** main(String[] args) {

**int** a = 1;

**do** {

**if** (a % 3 == 0 || a % 5 == 0) {

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

}

a++;

} **while** (a <= 100);

}

}

1. 对数组3、6、21、34、66、12、33进行求和。并输出结果。

For：

**package** com.test4;

**public** **class** Home3\_2 {

**public** **static** **void** main(String[] args) {

**int** sum=0;

**int** []a={3,6,21,34,66,12,33};

**for**(**int** i=0;i<=6;i++){

sum = sum+a[i];

}

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

}

}

While：

**package** com.test4;

**public** **class** Home3\_2 {

**public** **static** **void** main(String[] args) {

**int** sum = 0;

**int** i = 0;

**int**[] a = { 3, 6, 21, 34, 66, 12, 33 };

**while** (i <= 6) {

sum = sum + a[i];

i++;

}

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

}

}

do while:

**package** com.test4;

**public** **class** Home3\_2 {

**public** **static** **void** main(String[] args) {

**int** sum = 0;

**int** i = 0;

**int**[] a = { 3, 6, 21, 34, 66, 12, 33 };

**do** {

sum = sum + a[i];

i++;

} **while** (i <= 6);

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

}

}

1. 对数组45，67，39，5，16，20，10进行从大到小的冒泡排序。

For:

**package** com.test4;

**public** **class** Home3\_3 {

**public** **static** **void** main(String[] args) {

**int**[] a = { 45,67,39,5,16,20,10 };

**for**(**int** i=0;i<=a.length-1;i++){

**for**(**int** j=0;j<a.length-1-i;j++){

**if**(a[j]<a[j+1]){

**int** t=a[j];

a[j]=a[j+1];

a[j+1]=t;

}

}

}

**for**(**int** num:a){

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

}

}

}

While:

**package** com.test4;

**public** **class** Home3\_3 {

**public** **static** **void** main(String[] args) {

**int** i=0;**int** j=0;

**int**[] a = { 45, 67, 39, 5, 16, 20, 10 };

**while**(i<=a.length-1){

**while**(j<a.length-1-i){

**if**(a[j]<a[j+1]){

**int** t=a[j];

a[j]=a[j+1];

a[j+1]=t;

}

j++;

}

i++;

}

**for**(**int** num:a){

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

}

}

}

do while:

**package** com.test4;

**public** **class** Home3\_3 {

**public** **static** **void** main(String[] args) {

**int** i = 0;

**int**[] a = { 45, 67, 39, 5, 16, 20, 10 };

**do**{

**int** j = 1;

**do**{

**if**(a[j-1]<a[j]){

**int** t=a[j];

a[j]=a[j-1];

a[j-1]=t;

}

j++;

}**while**(j<a.length-i);

i++;

}**while**(i<=a.length);

**for**(**int** num:a){

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

}

}

}

1. 谈谈如何理解类和对象，及封装，继承和多态，并举例说明。
2. 类是具有相同属性和方法的一组对象的集合，类是对象的抽象，对象是类的具体。

人类属性有年龄体重身高等，行为衣食住行等，张三李四等每个人就是具体的对象。

1. 封装：将类的某些信息隐藏在类内部，不允许外部程序直接访问，而是通过该类提供的方法来实现对隐藏信息的操作和访问。

**public** **class** Pet {

**private** String name;

**private** **int** health;

**private** **int** love;

**public** String getName() {

**return** name;

}

**public** **int** getHealth() {

**return** health;

}

**public** **int** getLove() {

**return** love;

}

1. 继承：子类继承父类非private修饰的属性方法，继承的关键字是extends，访问父类的关键字是super

父类：

**public** **class** Pet {

**private** String name;

**private** **int** health;

**private** **int** love;

**public** **void** print() {

}

**public** String getName() {

**return** name;

}

**public** **int** getHealth() {

**return** health;

}

**public** **int** getLove() {

**return** love;

}

子类：

**public** **class** Dog **extends** Pet {

**private** String strain;

**public** String getStrain() {

**return** strain;

}

**public** **void** test() {

**this**.getName();

}

**public** Dog() {

**super**(" name");

}

**public** **void** setName(String string) {

}

**public** **void** run() {

}

}

1. 多态：三个必要条件是继承、重写、向上转型。

**class** A {

**public** String show(D obj) {

**return** ("A and D");

}

**public** String show(A obj) {

**return** ("A and A");

}

}

**class** B **extends** A { //B 继承A 后 重写了A的show(A)

//B有的方法show(A)show（B）show（D）

**public** String show(B obj) {

**return** ("B and B");

}

**public** String show(A obj) {

**return** ("B and A");

}

}

**class** C **extends** B { //有的方法show(A)show（B）show（D）

}

**class** D **extends** B {

}

**public** **class** Test {

**public** **static** **void** main(String[] args) {

A a1 = **new** A(); //a1--A,D

A a2 = **new** B();//向上转型 a2--B类里的A，D

B b = **new** B();//b--B里的方法A,B,D

C c = **new** C();//同上

D d = **new** D();//同上

System.***out***.println("1--" + a1.show(b));//show(B)在B类，是A的子类a1

System.***out***.println("2--" + a1.show(c));//CD同级

System.***out***.println("3--" + a1.show(d));//A类的方法

System.***out***.println("4--" + a2.show(b));//向上转型取相同的方法赋给父类

System.***out***.println("5--" + a2.show(c));//C的父类的父类是A，所以B and A；

System.***out***.println("6--" + a2.show(d));//D有对应方法，所以A and D

System.***out***.println("7--" + b.show(b));//继承，B的方法

System.***out***.println("8--" + b.show(c));//C 的父类是B，所以B and B

System.***out***.println("9--" + b.show(d));//D有相应方法，所以A and D

}

}

1. 在不同的package（包）下写2个类，一个类有4个属性或方法（例如1个属性，3个方法，或2个属性，2个方法），他们的作用域分别是private/public/不写(默认)/protected,从另一个类用main函数进行调用，观察下列情况：
2. 何时可以访问，或调用

Private只可以在当前类访问调用；public可以在当前类，同一包，子类，其他包访问调用；默认作用域可以在当前类，同一包访问调用；protected可以在当前类，同一包，子类访问调用。

1. 何时在当前类的package下面一行需要有import 操作

引用其他包里public，protected修饰的类方法属性时用import导入包。

1. 写3个类，类C3继承C2，C2类继承C1，在每个类的构造函数中加入this的输出，当子类构造函数运行时，每个父类的构造函数运行情况，及每个this之间关系，及与最后运行到的类对象obj之间的关系。

**package** com.test4;

**public** **class** C1 {

String name1;

**public** C1() {

String name = "a";

**this**.name1 = name;

System.***out***.println("C1" + name1);

}

}

**class** C2 **extends** C1 {

String name2;

**public** C2() {

**super**();

String name = "b";

**this**.name2 = name;

System.***out***.println("C2" + name2);

}

}

**class** C3 **extends** C2 {

String name3;

**public** C3() {

**super**();

String name = "c";

**this**.name3 = name;

System.***out***.println("C3" + name3);

}

}

测试结果：



父类C1运行打印C1a，C2运行时super先访问父类C1，打印C1a，然后打印C2b，C3运行时，先访问基类打印C1a，然后访问父类C2b，最后打印C3c。this调用自己类的属性。

1. 写3个类，分别是：1个抽象类，与2个实现类，抽象类中有一个抽象方法和一个具体方法，通过main函数测试抽象类的构造过程。

**package** com.test4;

**public** **abstract** **class** Home7 {

**public** **abstract** **void** a();

**public** **void** print() {

System.***out***.println("具体方法");

}

}

**class** H **extends** Home7 {

**public** **void** a() {

System.***out***.println("抽样方法的实现");

}

}

测试：

**package** com.test4;

**class** Home7test{

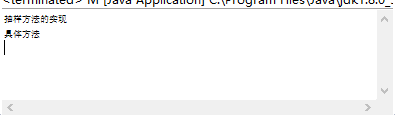
**public** **static** **void** main(String [] args){

H b=**new** H();

b.a();

b.print();

}

}

1. 写一个接口，接口中有1――2个方法，然后写2个实现类，体会接口的用法。

**package** com.test4;

**public** **interface** Home8 {

**public** **void** a();

**public** **void** b();

}

**class** U **implements** Home8 {

**public** **void** a() {

System.***out***.println("1");

}

**public** **void** b() {

System.***out***.println("2");

}

}

测试：

**package** com.test4;

**import** org.junit.Test;

**public** **class** Home8test{

@Test

**public** **void** test(){

U test1=**new** U();

test1.a();

test1.b();

}

}

Home8接口里有两个抽象方法a（），b（），类U实现接口Home8，分别赋予打印1，2；

测试类实例化类U可以引用方法a（），b（）。