**第 二 次 大 作 业**

**一、阅读程序写结果**

**1.**

public class SubClass extends SuperClass {

int x = 5;

SubClass() {

System.out.println("construct SubClass");

}

void work() {

super.work();

System.out.println("SubClass.work() ");

System.out.println("super.x=" + super.x + ",sub.x=" + x);

}

public static void main(String[] args) {

SubClass subC = new SubClass();

subC.work();

}

}

class SuperClass {

int x;

SuperClass() {

System.out.println("construct SuperClass");

}

void work() {

System.out.println("SuperClass.work() ");

}

}

**2.**

import java.util.\*;

public class TestSet {

public static void main(String arg[]) {

Set<Point> set = new TreeSet<Point>();

set.add(new Point(2, 2));

set.add(new Point(1, 1));

set.add(new Point(1, 5));

set.add(new Point(0, 0));

set.add(new Point(5, 1));

System.out.println(set.size());

for (Point s : set)

System.out.println(s.x + "," + s.y);

}

}

class Point implements Comparable<Point> {

int x = 0, y = 0;

public Point(int x, int y) {

this.x = x;

this.y = y;

}

public int compareTo(Point o) {

if (x > o.x)

return 1;

else if (x < o.x)

return -1;

else

return 0;

}

}

**3.**

import java.io.FileInputStream;

import java.io.FileOutputStream;

public class FileTest {

public static void main(String[] args) throws Exception {

int data1 = -2;

FileOutputStream fout = new FileOutputStream("f1.txt");

fout.write(data1);

fout.write(new byte[] { -1, -2 });

fout.close();

FileInputStream fin = new FileInputStream("f1.txt");

int data2 = fin.read();

byte[] b = new byte[4];

int count = fin.read(b);

fin.close();

for (int i : b) {

System.out.println(i);

}

System.out.println(count);

System.out.printf("data1=%d,data2=%d",data1,data2);

}

}

**4.**

public class CallMe {

public static void main(String[] args) {

mynumber(12345);

}

public static void mynumber(int n) {

System.out.print(n % 10);

if (n / 10 > 0) {

mynumber(n / 10);

} else {

System.out.println("\nA");

}

System.out.print(n % 10);

}

}

**5.**

public class RevString {

public static void main(String[] args) {

String str1 = new RevString().rev1("ABC");

System.out.println(str1);

String str2 = new RevString().rev2(str1);

System.out.println(str2);

}

public String rev1(String s) {

if (s.length() == 0)

return s;

else {

String tmp = rev1(s.substring(1)) + s.substring(0, 1);

System.out.println(tmp);

return tmp;

}

}

public String rev2(String s) {

String s2 = "";

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

s2 = s.charAt(i) + s2;

}

return s2;

}

}

**6.**

public class ForTest {

public static void main(String[] args) {

int k = 0;

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

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

System.out.print(i);

k++;

}

System.out.println();

}

System.out.printf("k=%d\n", k);

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

for (int j = 3 - i; j >= 1; j--)

System.out.print(i);

System.out.println();

}

}

}

**7.**

import java.util.ArrayList;

import java.util.List;

import java.util.Set;

import java.util.TreeSet;

public class TestList {

public static void main(String[] args) {

List<Integer> nums = new ArrayList<Integer>();

nums.add(0);

nums.add(1);

nums.add(2);

nums.add(3);

nums.add(4);

nums.add(5);

nums.add(6);

nums.add(7);

nums.add(8);

System.out.println(nums);

Set<Integer> s = new TreeSet<Integer>();

for (int i = 0; i < nums.size(); i++) {

s.add(nums.remove(i));

}

System.out.println(nums);

nums.addAll(s);

System.out.println(nums);

}

}

**8.**

class OverrideTest {

void show() {

System.out.println("super show");

System.out.println(this.getClass().getName());

this.getName();

}

void getName() {

System.out.println("OverrideTest");

}

}

public class SubOverride extends OverrideTest {

void show() {

System.out.println("sub show");

super.show();

this.getName();

}

void getName() {

System.out.println(this.getClass().getSuperclass().getName());

}

public static void main(String args[]) {

OverrideTest s = new SubOverride();

s.show();

}

}

**9.程序无异常情况发生的执行结果**

public class ThreadTest {

public static void main(String[] args) throws Exception {

Thread t1 = new Thread(new SimpleRunner());

System.out.println("main begins");

t1.start();

t1.join();

System.out.println("\nin main");

Thread t2 = new Thread(new Runnable() {

public void run() {

for (int i = 5; i < 9; i++) {

System.out.print(i);

}

System.out.println("\nin t2");

}

});

t2.start();

t2.join();

System.out.println("main ends");

}

}

class SimpleRunner implements Runnable {

public void run() {

System.out.println("in simple");

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

System.out.print(i);

}

}