

**一、 单项选择题（每小题1分，共26分）**

**1. 下面哪个会产生语法错误 （ D ）**

A．int Marks[]; B．int[] Marks;

C. int Marks[]={0,1,2,3}; D. int Mark[4];

选项 D. **int Mark[4];** 会产生语法错误。

在Java中，声明数组时，方括号 [] 的位置可以放在类型名称后面（选项 A 和 B），或者直接放在变量名后面（选项 C）。这两种方式都是合法的。

选项 D 中，将方括号放在变量名 **Mark** 后面，而不是类型名称 **int** 后面，这是不允许的。在声明数组时，方括号 [] 应该紧跟在类型名称之后。

正确的写法应该是 **int[] Mark = new int[4];**，其中 **new int[4]** 用于创建一个包含四个元素的整数数组。

**2. 下面代码产生多少个星号？ ( C )**

for(int i=4,i<=20; i++)

System.out.println(“\*”);

A. 15 B. 16 C. 17 D. 19

**3. 下面代码产生多少个星号？( B )**

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

for(int j=i; j>=0; j--)

System.out.print(”\*”);

A. 15 B. 21 C. 6 D. 20 E. 16

外层循环执行 6 次（i=0,1,2,3,4,5）。 当 i=0 时，内层循环执行 1 次（j=0）。 当 i=1 时，内层循环执行 2 次（j=1,0）。 当 i=2 时，内层循环执行 3 次（j=2,1,0）。 当 i=3 时，内层循环执行 4 次（j=3,2,1,0）。 当 i=4 时，内层循环执行 5 次（j=4,3,2,1,0）。 当 i=5 时，内层循环执行 6 次（j=5,4,3,2,1,0）。

所以，总共会打印 1+2+3+4+5+6=21 个星号。

**4.下面代码的输出结果是：（ D ）**

int score = 80; int x=100;

String type = score<60?”不及格”:”及格”;

int flag=x>0?1:(x==0?0:-1);

system.out.print(type+” “);

system.out.println(flag);

A． 及格 -1 B：编译出错 C． 及格 0

D. 及格 1 E． 不及格 1 F. 不及格 0

**5. 下面代码的输出结果是：（ B ）**

int i; int j=1;

while(j>0){

System.out.print("hello ");

j--;

i=3;

}

System.out.println(i);

A. 编译错误 B. hello 3

C. hello hello 3 D. hello 0

**6. 哪个关键字可以对对象加互斥锁？ ( B )**

A. transient B. synchronized

C. serialize D. Static

7. 为保护本地主机，对Applet安全限制中正确的是 ( )

A. Applet可加载本地库或方法

B. Applet可读、写本地计算机的文档系统    C. Applet可向Applet之外的任何主机创建网络连接

D. 没有被授权，Applet不能运行任何可执行程序

8. JApplet默认的布局方式是 ( )

A. BorderLayout B. FlowLayout

C. Null D. GridLayout

**9. 下面的哪些赋值语句是不对的( A )**

A. float f = 11.1; B. double d = 5.3E12; C. double d = 3.14159; D. double d = 3.14D.

A. **float f = 11.1;** 会产生编译错误。因为在Java中，浮点数默认为 **double** 类型，所以要将一个浮点数直接赋给 **float** 类型的变量时，需要在数值后面加上 **f** 或 **F** 后缀，表示这是一个 **float** 类型的字面值。正确的写法应该是 **float f = 11.1f;**。

**10. 给出下面的代码**

1) public void modify() {

2)    int i, j, k;

3)    i = 100;

4)    while ( i > 0 ) {

5)       j = i \* 2;

6)       System.out.println (" The value of j is " + j );

7)       k = k + 1;

8)       i--;

9)    }

10) }

哪一行在编译时可能产生错误( C )

A. line 4 B. line 6 C. line 7 D. line 8

代码中的第 7 行 **k = k + 1;** 尝试对变量 **k** 进行增加操作，但是在此之前并没有给变量 **k** 进行初始化。在Java中，局部变量需要在使用之前进行初始化。由于变量 **k** 在第 2 行声明，但没有进行初始化操作，所以在第 7 行进行 **k = k + 1;** 的操作时会产生编译错误。

**11. 给出下面的代码**

public class Person{

       static int arr[] = new int[10];

       public static void main(String a[]) {

          System.out.println(arr[1]);

       }

    }

**以下哪个叙述是对的 ( C )**

A.      编译时将发生错误。 B.      编译时正确但是运行时出错。

C.      输出为0。 D.     输出为null

**12. 给出下面的代码**

public class Person{

String name=“unknown”;

public static void main(String args[]){

System.out.println(Person.name);

}

}

**以下哪个叙述是对的( A )**

A.      编译时将发生错误。 B.      编译时正确但是运行时出错。

C.      输出字符串“unknown” D.     输出为null

name需要static

**13. 一个类中定义的成员变量只能被同一包中的类访问。下面的哪些修饰符可以获得需要的访问控制( B )**

A. private B.  no modifier C. public D. protected

**14. 方法resume()负责恢复哪些线程的执行（ D ）**

A. 通过调用stop()方法而停止的线程。 B. 通过调用sleep()方法而停止运行的线程。

C. 通过调用wait()方法而停止运行的线程。 D. 通过调用suspend()方法而停止运行的线程。

在早期的 Java 版本中，可以使用 **suspend()** 方法暂停一个线程的执行，该方法使线程进入挂起状态。然而，该方法已经被标记为不安全，不推荐使用，因为它可能导致线程死锁和其他并发问题。

为了避免潜在的问题，如果使用了 **suspend()** 方法暂停了一个线程，那么可以使用 **resume()** 方法来恢复该线程的执行，使其继续运行。

所以，方法 **resume()** 负责恢复通过调用 **suspend()** 方法而停止运行的线程。

**15. 下面哪个不是InputStream类中的方法 （ B ）**

A. int read(byte[]) B. void flush() C. void close() D. int available()

A. **int read(byte[])** 用于从输入流中读取字节，并将其存储到字节数组中。

C. **void close()** 用于关闭输入流，释放与之相关的资源。

D. **int available()** 用于返回可以从输入流中读取而不阻塞的字节数。

这些方法都属于 **InputStream** 类的方法。

然而，选项 B. **void flush()** 并不是 **InputStream** 类中的方法。该方法通常用于刷新输出流的缓冲区，而不是读取输入流的数据。

16. 下面哪个不能被添加到容器中？( )

A. an Applet B. a Component C. a Container D. a Menu

**17. 关于以下程序代码的说明正确的是（ D ）**

1． class StaticTest{

2． private static int x=100；

3． public static void main(String args[ ]){

4． StaticTest s=new StaticTest ( );

5． s.x++;

6． StaticTest s2=new StaticTest ( );

7． s2.x++;

8． s=new StaticTest ( );

9． s.x++;

10． StaticTest.x--;

11． System.out.println(“x=”+x);

12． }

13． }

A、 5行不能通过编译，因为引用了私有静态变量

B、 10行不能通过编译，因为x是私有静态变量

C、 程序通过编译，输出结果为：x=103

D、 程序通过编译，输出结果为：x=102

**18. 类Test1定义如下：**

1．public class Test1{

2． public float aMethod（float a，float b）{ }

3．

4．}

**将以下哪种方法插入行3是不合法的。( B )**

A、 public float aMethod（float a， float b，float c）{ }

B、 public float aMethod（float c，float d）{ }

C、 public int aMethod（int a， int b）{ }

D、 private float aMethod（int a，int b，int c）{ }

**19. 关于以下程序段的说法，正确的是（ A ）**

1. class MyListener implements

2． ActionListener，ItemListener{

3． public void actionPerformed（ActionEvent ae）{

4． System.out.println("Action");}

5． public void itemStateChanged(ItemEvent ie){

6． System.out.println("Item");

7． }

8． }

A、 可通过编译

B、 第2行产生编译错误

C、 第3行产生编译错误

D、 第5行产生编译错误

**20. 关于以下代码所画图形的说明，正确的是（ B ）**

g.setColor(Color.black);

g.drawLine(10,10,10,50);

g.setColor(Color.red);

g.drawRect(100,100,150,150);

A、 一条40像素长的垂直红线，一个边长为150像素的红色四方形

B、 一条40像素长的垂直黑线，一个边长为150像素的红色四方形

C、 一条40像素长的垂直黑线，一个边长为50像素的红色四方形

D、 一条50像素长的垂直红线，一个边长为150像素的黑色四方形

21. class A

{ int x;

public static void main(String [] agrs)

{ A a=new A();

a.x=5;

change(a);

System.out.println(a.x);

}

public static void change(A a)

{ a.x=3;

}

}

**上面代码的输出结果为：（ A ）**

A. 3 B. 5 C. 2 D. 不是以上的值

**22. 假设下列的命令能正确编译Cat.java，对这个命令的解释正确的是 ( B )**

javac –d c:\temp Cat.java

A. 源代码文档放在c:\temp 目录下，产生的字节码文档也放在c:\temp 目录下。

B. 源代码文档放在当前目录下，产生的字节码文档放在c:\temp 目录下。

C. 源代码文档放在c:\temp 目录下，产生的字节码文档放在当前目录下。

D. 需要手动在c目录下创建一个temp文档夹，再将字节码文档从当前目录拷贝一份到c:\temp 目录下。

23.

class A {  
 A() {  
 System.*out*.print("A constructor ");  
 }  
 void cry() {  
 System.*out*.print("You are right!");  
 }  
}  
class B extends A {  
 B() {  
 System.*out*.print("B constructor ");  
 }  
  
 void cry() {  
 System.*out*.print("You are wrong!");  
 }  
}  
class Example {  
 public static void main(String args[]) {  
 A a = new B();  
 a.cry();  
 }  
}

**假设上面的代码都放在同一个文档Example.java中，对上面代码的解释正确的是 ( A )**

A．编译不能通过

B. 输出结果为“You are wrong!”

C. 输出结果为“A constructor You are wrong!”

D. 输出结果为“A constructor B constructor You are wrong!”

24.

byte d[ ]= "你我他".getBytes();

System.out.print(d.length + “ “);

String s=new String(d);

System.out.println(s.length());

上面代码的输出结果为：( **9 3** )

A. 6 3 B. 3 6

C. 8 4 D. 编译出错

由于 Java 中使用 UTF-8 编码，每个中文本符占用 3 个字节，所以 **"你我他"** 这个字符串将被转换为一个长度为 9 的字节数组。

然后，使用 **System.out.print()** 打印了字节数组 **d** 的长度，即 **d.length**，输出结果为 9。

接下来，通过创建一个新的字符串 **s**，使用 **new String(d)** 将字节数组转换为字符串。由于 UTF-8 编码中每个中文本符占用 3 个字节，所以字节数组的长度为 9，但字符串的长度仅为 3，因为它包含了 3 个中文本符。

最后，使用 **System.out.println()** 打印了字符串 **s** 的长度，即 **s.length()**，输出结果为 3。

25.

StringBuffer sb1 = new StringBuffer("a");

StringBuffer sb2 = new StringBuffer("a");

System.out.print( sb1.equals(sb2));

System.out.println( sb1==sb2);

**以上代码运行结果是：( D )**

A. true false B. false true C.true true D. false false

**26. 如果试图编译和运行下面的程序，将会产生什么输出？ ( A )**

public class Rand{

public static void main(String args){

int iRand;

iRand = Math.random();

System.out.println(iRand);

}

}

A. 编译错误 B. 输出1到10之间的一个整数

C. 输出0或者1 D. 输出0.0到1.0之间的一个数

在 **main** 方法的参数中，使用了错误的类型 **String args**，正确的类型应该是 **String[] args**。所以编译器会报错，无法编译成功。

**Math.random()** 方法返回的是一个 **double** 类型的随机数，而 **iRand** 声明为 **int** 类型，无法直接将 **double** 类型的值赋给 **int** 类型的变量。

**二．多项选择题（以下题目含两个以上答案）（每小题2分，共16分）**

1. 下面关于变量及其范围的陈述哪些是对的。 （ acd ）

A.      实例变量是类的成员变量。 B.      实例变量用关键字static声明。 C.      在方法中定义的局部变量在该方法被执行时创建 D.     局部变量在使用前必须被初始化。

2. 有关线程的哪些叙述是对的 ( bd )

　A. 一旦一个线程被创建，它就立即开始运行。

　B. 使用start()方法可以使一个线程成为可运行的，但是它不一定立即开始运行。

　C. 当一个线程因为抢先机制而停止运行，它被放在可运行队列的前面。

　D. 一个线程可能因为不同的原因停止并进入就绪状态。

3.

String s= "hello";

String t = "hello";

char c[] = {’h’,’e’,’l’,’l’,’o’} ;

以下哪些返回true？ （ acd ）

A. s.equals(t); B. t.equals(c);

C. s==t; D. t.equals(new String("hello"));

E. t==c.（不可比较）



4.

public class Parent{

int change() {….}

}

class Child extends Parent{

}

哪些方法可以被加入类Child ( abd )

A. public int change(){} B. int chang ( int i){}

C. private int change() {} D. abstract int chang() {}

在类 **Child** 中可以加入以下方法：

A. **public int change(){}**：子类可以重写父类的非私有方法，并且访问权限可以比父类方法更宽松（如从默认访问权限变为 **public**）。

B. **int chang(int i){}**：子类可以添加自己的方法，与父类的方法没有关系。

D. **abstract int chang(){}**：子类可以声明抽象方法，但需要实现该抽象方法。

因此，选项 A、B 和 D 可以被加入类 **Child**。选项 C 是私有方法，不可继承或重写。

5. 下面关于继承的哪些叙述是正确的。（ ad ）

　　 A． 在Java中只允许单一继承。

　　 B． 在Java中一个类只能实现一个接口。

　　 C． 在Java中一个类不能同时继承一个类和实现一个接口。

　　 D． Java的单一继承使代码更可靠。

6. 下列哪些部件(Component)会产生ActionEvent （ ）

A. JButton B. JLabel C. JMenuItem D. JTextField

7. 哪些类可以实例化读取一个字符流？ ( bcd )

A. InputStream B. InputStreamReader

C. FileReader D. BufferedReader

B. InputStreamReader：用于将字节流转换为字符流，可以指定字符集编码。

C. FileReader：用于读取字符文档，底层基于 InputStreamReader 实现。

D. BufferedReader：用于缓冲字符输入流，提供了更高效的读取方法。

因此，选项 B、C 和 D 都可以实例化读取一个字符流。选项 A 的 InputStream 是字节流类，不能直接用于读取字符流。

8. public void test() {

       try { oneMethod();

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

       } catch (ArrayIndexOutOfBoundsException e) {

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

      } catch(Exception e) {

            System.out.println("condition 3");

} finally {

            System.out.println("finally");

      }

    }

**在oneMethod()方法运行正常的情况下将显示什么？( ad )**

A. condition 1 B. condition 2 C. condition 3 D. finally

**三．程序阅读题。（共 18 分）**

1. 以下程序段的输出结果为 （3分） txa

class T {

T() {

System.out.print("t ");

}

}

class X extends T {

X() {

System.out.print("x ");

}

}

class A extends X {

A() {

System.out.print("a ");

}

}

class OrderOfConstruction {

public static void main(String args[]) {

A pine = new A();

}

}

2. 以下程序段的输出结果为 （3分）int version

class A{

void test( int i ){

System.out.println("int version");

}

void test(String s){

System.out.println("String version");

}

public static void main(String args[ ]){

A a=new A ( );

char ch='p';

a.test(ch);

}

}

3. 已知类 MyQuestion 和 类MyProblem 定义如下：

public class MyQuestion{

protected int a;

protected int b;

public MyQuestion(){

this.a =0;

this.b = 0;

}

public MyQuestion( int x, int y){

this.a = x;

this.b = y;

}

public MyQuestion( int x){

this();

this.a = x;

}

public int enquire(){

return this.a + this.b;

}

public int interrogate(int x){

return (this.a + this.b)\*x;

}

public void display(){

System.out.println(this.a + " " + this.b);

}

}

public class MyProblem extends MyQuestion{

protected int c;

public MyProblem(){

this.c = 0;

}

public MyProblem(int x, int y, int z){

super(y,z);

this.c = x;

}

public MyProblem(int x, int y){

super(x,y);

}

public MyProblem( int x){

this.c = x;

}

public int enquire(){

this.c = this.a \* this.b;

return this.c;

}

public void display(){

System.out.println(this.a + " " + this.b + " " + this.c);

}

}

写出以下代码的输出结果：

MyQuestion q; q = new MyQuestion(1);

System.out.print(q.interrogate(9) + " ");

q.display();

输出结果 （3分）

MyProblem p;

p= new MyProblem(20,2);

System.out.println(p.enquire());

System.out.print(p.interrogate(2) + " ");

p.display();

输出结果 （3分）

MyProblem p;

p = new MyProblem(20,2,4);

p.display();

p.enquire();

p.display();

输出结果 （3分）

MyProblem p;p= new MyProblem(10);

MyQuestion q = new MyQuestion(10);

System.out.print(p.enquire() + " ");

p.display();

System.out.print(q.enquire() + " ");

q.display();

输出结果 （3分）

**四． 编程题：（共 40 分）按以下要求编写完整的应用程序，附录中有提供某些类，接口的构造函数和方法（没有提供全部的类和方法）仅供参考！**

1. ( 20 分 ) 编写应用程序播放音频文档，当用户选择下列列表中的一个项目后，按“play“按钮可以播放选中的音乐；按“loop”开始播放并且循环播放；按“stop”按钮停止播放。假设目前只有3首歌曲：song1.au,song2.au,song3.au 并且这3首歌固定放在当前目录下的一个songs的子目录下。界面的布局可以自由设定。

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

public class AudioPlayerApp extends JFrame {

private JButton playButton;

private JButton loopButton;

private JButton stopButton;

private JComboBox<String> songList;

public AudioPlayerApp() {

setTitle("Audio Player");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new FlowLayout());

// 创建音乐列表下拉框

String[] songs = {"song1.au", "song2.au", "song3.au"};

songList = new JComboBox<>(songs);

add(songList);

// 创建播放按钮

playButton = new JButton("Play");

add(playButton);

// 创建循环按钮

loopButton = new JButton("Loop");

add(loopButton);

// 创建停止按钮

stopButton = new JButton("Stop");

add(stopButton);

// 为按钮添加事件监听器

playButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedSong = (String) songList.getSelectedItem();

playAudio(selectedSong);

}

});

loopButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedSong = (String) songList.getSelectedItem();

loopAudio(selectedSong);

}

});

stopButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

stopAudio();

}

});

pack();

setVisible(true);

}

// 播放音频文档

private void playAudio(String filename) {

try {

String filepath = "songs/" + filename; // 假设音频文档在songs子目录下

File file = new File(filepath);

// 调用播放音频的代码

System.out.println("Playing: " + filename);

} catch (Exception e) {

e.printStackTrace();

}

}

// 循环播放音频文档

private void loopAudio(String filename) {

try {

String filepath = "songs/" + filename; // 假设音频文档在songs子目录下

File file = new File(filepath);

// 调用循环播放音频的代码

System.out.println("Looping: " + filename);

} catch (Exception e) {

e.printStackTrace();

}

}

// 停止播放音频文档

private void stopAudio() {

// 调用停止播放音频的代码

System.out.println("Stopping audio");

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

new AudioPlayerApp();

}

});

}

}

2. 基于UDP的字符串传输：( 20 分 )

编写主机1和主机2的应用程序，主机1使用DatagramSocket对象将字符串“厦门大学软件学院”发送到主机2，主机2将数据每隔一秒逐字显示在窗口上。显示的字体，颜色可以任意设定。主机2的IP地址可以假定为：127.0.0.1

主机1（发送方）代码：

import java.net.\*;

import java.io.\*;

public class Host1 {

public static void main(String[] args) {

try {

String message = "厦门大学软件学院";

InetAddress address = InetAddress.getByName("127.0.0.1");

int port = 5000;

DatagramSocket socket = new DatagramSocket();

byte[] buffer = message.getBytes();

DatagramPacket packet = new DatagramPacket(buffer, buffer.length, address, port);

socket.send(packet);

System.out.println("数据已发送");

socket.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

主机2（接收方）代码：

import java.net.\*;

import java.io.\*;

public class Host2 {

public static void main(String[] args) {

try {

int port = 5000;

DatagramSocket socket = new DatagramSocket(port);

byte[] buffer = new byte[1];

DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

while (true) {

socket.receive(packet);

String message = new String(packet.getData(), 0, packet.getLength());

System.out.print(message);

Thread.sleep(1000);

}

} catch (Exception e) {

e.printStackTrace();

}

}

为了实现每秒显示一个字符的效果，我将接收缓冲区的大小设置为1(中文可能要改成3)，并使用Thread.sleep(1000)来暂停一秒钟。这样可以保证每秒只显示一个字符。

以下是在同一个窗口中每秒的运行效果图。

附录：

1. JComboBox

2. JFrame

3 ActionListener,

4 ItemListener

5.Runnable

6. Thread

7．DatagramPacket

8. DatagramSocket