数学学院本科生 2020-2021 学年第 2 学期 Java 课程 期末考试试卷（A 卷）

专业：＿＿＿＿＿＿＿＿＿＿年级：＿＿＿＿＿学号：＿＿＿＿＿＿＿姓名：＿＿＿＿＿＿＿＿

题目	一	二	三	四	五	六	卷面成绩
分数							

一．单项选择题（40 分，每小题 4 分）

	1. char 类型的变量可以被赋值的范围是：	**A.** 0 through 2^32-1	**B.** 0 through 2^15-1	**C.** 0 through 2^16-1	**D.** -2^15 through 2^15-1
	2.如果 char test[]=new char[10],则，test[2]=?	**A.** 10	**B.** 0	**C.** null	**D.** ‘\u0000’
	3. The Java keyword that says class B adds something to, or builds upon the existing definition for class A is ?	**A.** extends	**B.** import	**C.** derives	**D.** inherits
	4.下面哪些代码会使 x 的值为 0：	**A.** int x = 8; x = x>>>3;	**B.** int x = 16; x = x <<4 ;	**C.** int x = -1; x = x >>>1 ;	**D.** int x =-1 ; x = x >>1;
	5. Suppose x is 5 and y is 7. What is the value of the following expression: (x != 7) && (x <= y)	**A.** This is an invalid expression in Java.	**B.** None of these	**C.** true	**D.** false
	6.In the inheritance hierarchy Animal, Mammal, Rodent, "Mickey Mouse", Mammal is a ＿＿＿＿＿＿ of Animal.?	**A.** generalization	**B.** ancestor	**C.** subclass	**D.** superclass
	7. You are ＿＿ required to write a constructor method for a class.?	**A.** often	**B.** never	**C.** always	**D.** sometimes
	8. int a = 21 ; int b = 70；char c = ‘\u0070’; double f = 21.0;	**A.** a >= c	**B.** b > c	**C.** c <= b	**D.** c > f
	9. Which of the following methods prints a list of the methods that were called before the exception was thrown?	**A.** getMessages( )	**B.** printCalledMethods( )	**C.** traceMethodStack( )	**D.** printStackTrace( )
	10.下面哪个修饰符能防止相同包中的子类实现覆盖（override）:	**A.** public	**B.** private	**C.** final	**D.** static

二．判断题（15 分，每小题 1.5 分）

□ 1.如果 Thread 类的某实例中的 run 方法的返回值为-1,则该线程没有成功执行。

□ 2.循环语句块中的 continue 语句会终止整个循环语句的执行。

□ 3.一般而言，Try 语句后面不一定总是跟着 finally 语句。

□ 4.循环条件可以为数值变量，可以这样使用：int x=3; while (x) { x -- ; }

□ 5.以 static 修饰的成员变量，叫类变量；否则，叫实例变量。

□ 6.类变量的生存期不依赖于对象，相当于 C 语言中全局变量的作用。

□ 7. final 修饰变量，变量就变成了常量；final 修饰方法，方法就不能再覆盖。

□ 8.用 abstract 关键字来修饰一个类时，这个类叫做抽象类。

□ 9.只能在定义的类中使用内部类，内部类可以访问类变量和实例变量。

□ 10.在 JDK1.0 的 AWK 中事件处理模型是委托模型，委托模型是基于容器的。

三．简答题(请写出右边代码的输出并解释）（ 15 分）

```
class Hotel {
    private int NO_OF_ROOMS = 12;
    private int MAX_NO_OF_GUEST = initMaxGuests();
    private int OCCUPANCY_PER_ROOM = initOccupancy();
    private int initMaxGuests() {
        System.out.println("Occupancy_PER_ROOM:          "          +
OCCUPANCY_PER_ROOM );
        System.out.println("MAX_NO_OF_GUEST: " + NO_OF_ROOMS *
                OCCUPANCY_PER_ROOM);
        return NO_OF_ROOMS * OCCUPANCY_PER_ROOM;  }
    public int getMaxGuests() {    return MAX_NO_OF_GUEST;    }
    public int initOccupancy() {  return 2;     }
    public int getOccupancy() {  return OCCUPANCY_PER_ROOM;     }
}
public class TestOrder {
    public static void main(String args[]) {
        Hotel objRef = new Hotel();
        System.out.println("After object creation: ");
        System.out.println("OCCUPANCY_PER_ROOM:          "          +
objRef.getOccupancy());
        System.out.println("MAX_NO_OF_GUEST:  "  +  objRef.getMaxGuests());
    }
}
```

四、阅读右边 67 行代码，回答下面的问题。(共 10 分)

4.1 请解释变量 blinkState 的意义 (3 分):

____

____

____

____

____

____

____

____

____

____

4.2 请列出事件监听器及事件响应过程?(4 分)

____

____

____

____

____

____

____

____

____

4.3 第 59 行的 new 起什么作用?(3 分)

____

____

____

____

____

____

____

____

____

____

```
1   import java.awt.*;   import java.awt.event.*;
2   import javax.swing.*;
3   public class HelloJava4 extends JComponent
4       implements  MouseMotionListener,  ActionListener,
    Runnable {
5
6     int messageX = 125, messageY = 95;
7     String theMessage;
8     JButton theButton;
9
10    int colorIndex;
11    static Color[ ] someColors = { Color.black, Color.red,
12          Color.green, Color.blue, Color.magenta };
13
14    boolean blinkState;
15
16    public HelloJava4(String message) {
17      theMessage = message;
18      theButton = new JButton("Change Color");
19      setLayout(new FlowLayout( ));
20      add(theButton);
21      theButton.addActionListener(this);
22      addMouseMotionListener(this);
23      Thread t = new Thread(this);
24      t.start( );
25    }
26    public void paintComponent(Graphics g) {
27      g.setColor(
        blinkState ? getBackground() : currentColor( )  );
29      g.drawString(theMessage, messageX, messageY);
30    }
31    public void mouseDragged(MouseEvent e) {
32      messageX = e.getX( ); messageY = e.getY( );
33      repaint( );
34    }
35  public void mouseMoved(MouseEvent e) {}
36    public void actionPerformed(ActionEvent e) {
37      if (e.getSource( ) == theButton)     changeColor( );
38    }

39    synchronized private void changeColor( ) {
40      if (++colorIndex == someColors.length)
41        colorIndex = 0;
42      setForeground(currentColor( ));
43      repaint( );
44    }
45    synchronized private Color currentColor( ) {
46      return someColors[colorIndex];     }
47
48    public void run( ) {
49      try {
50        while(true) {
51          blinkState = !blinkState;
52          repaint( );
53          Thread.sleep(500);
54        }
55      } catch (InterruptedException ie) { }
56    }
57    public static void main(String[] args) {
58      JFrame f = new JFrame("HelloJava4");
59      f.addWindowListener( new WindowAdapter( ) {
60        public void windowClosing(WindowEvent we)
61        { System.exit(0); }
          });
62      f.setSize(300, 300);
63      f.getContentPane( ).add(
64            new HelloJava4( "Hello, Java!" ) );
65      f.setVisible(true);
66    }
67  }
```