

武汉大学计算机学院

2019-2020 学年度第 1 学期 2018 级

《面向对象语言程序设计》期末考试试卷 A 卷答案

姓名：_____ 学号：_____ 专业：_____

说明：开卷考试，答案请全部写在答题纸上，写在试卷上无效。

未经主考教师同意，考试试卷、答题纸、草稿纸均不得带离，否则视为违规。

题号	一	二	三	四	总分
总分	30	20	20	30	100

一. 简答题（共 5 小题，共 30 分。每小题 6 分）

1、简述 Java 为什么能跨平台？

答：因为 Java 程序编译之后的代码不是能被硬件系统直接运行的代码，而是一种“中间码”——字节码。然后不同的硬件平台上安装有不同的 Java 虚拟机(JVM)，由 JVM 来把字节码再“翻译”成所对应的硬件平台能够执行的代码。因此对于 Java 编程者来说，不需要考虑硬件平台是什么。所以 Java 可以跨平台。

2、简述 java 类的静态成员和动态成员中的区别？

答：（1）所属不同：静态变量属于类，成为类变量；成员变量属于对象，称为对象变量；
（2）内存中位置不同：静态变量位于方法区中的静态区成员变量存储于堆内存；
（3）内存出现时间不同：静态变量随着类的加载而加载，随着类的消失而消失成员变量随着对象的创建而存在；
（4）调用不同：静态变量可以通过类名调用，也可以通过对象名调用成员变量只能通过对象名调用。

3、Java 中如何实现方法的覆盖？方法覆盖有哪些注意事项？

答：如果在子类中重新定义了一个与它的超类完全同名的方法，子类的方法将覆盖超类中的同名方法。

方法的覆盖中需要注意：子类在重新定义父类已有的方法时，应保持与父类完全相

同的方法头部声明，即应保持与父类有完全相同的方法名、返回类型和参数列表，否则就不是方法的覆盖，而是子类定义了一个与父类无关的方法，父类的方法仍然存在。另外，在访问权限上，只能放宽，否则编译时会出错。

4、简述 java 的内存回收机制？

答：java 内存回收采用垃圾回收机制。所谓垃圾回收机制是一种动态存储管理技术，它自动地释放不再被程序引用的对象，按照特定的垃圾收集算法来实现资源自动回收的功能。

5、解释方法的值传递和引用传递？

答：如果方法参数是基本数据类型，该参数采用值传递。值传递复制给形式参数的是一个值，复制之后，形式参数与实际参数就没有任何关系了。方法执行中形式参数值的改变不影响实际参数的值。

如果方法参数是引用数据类型，该参数采用引用传递。引用传递复制的是对象的地址副本而不是对象的数据。如果通过操作副本引用的值，修改了引用地址的对象，此时方法以外的引用此地址对象就会被修改。

二、程序阅读题（共 3 小题，共 20 分。第 1 题 6 分，第 2 题 7 分，第 3 题 7 分）

1、阅读下列程序，写出程序执行结果。

```
public class Test {  
    public static void main(String[] args) {  
        List list = new ArrayList();  
        list.add("35.26");  
        list.add(35.26);  
        list.add(50);  
        list.add(true);  
        for (int i = 0; i < list.size(); i++) {
```

```

        Object item = list.get(i);
        System.out.println(item.getClass());
    }
}
}

```

答案:

```

class java.lang.String
class java.lang.Double
class java.lang.Integer
class java.lang.Boolean

```

2、阅读下列程序，写出程序执行结果。

```

class A{
    double f(double x, double y) {
        return x + y;
    }
}
class B extends A{
    double f(int x , int y) {
        return x * y;
    }
}
public class Demo {
    public static void main(String args[]) {
        B b= new B();
        System.out.println(b.f(5,3));
        System.out.println(b.f(5.0,3.0));
    }
}

```

答: 15.0

8.0

3、阅读下列程序，写出程序执行结果

```
public class DivideZeroPractice {  
    public static void main(String[] args) {  
        int y;  
        try{  
            y=3/0;  
            System.out.println("execute divide ok!");  
        }  
        catch(ArithmeticException e){  
            System.out.println("divide by zero error!");  
        }  
        finally  
        {  
            System.out.println("execute finally!");  
        }  
        System.out.println("program ends ok!");  
    }  
}
```

答：输出结果：

divide by zero error!
execute finally!
program ends ok!

三、程序实现题（共 2 小题，共 20 分，每题 10 分）

1、请编写一个程序，产生 10 个介于[1, 100)的随机整数(包括 1,但不包含 100)。

答：代码参考如下

《面向对象语言程序设计》期末考试试卷 A 卷答案

4 / 11

```

public static void main(String[] args) {
    List list = new ArrayList();
    while (list.size() < 10) {
        int i = (int) (1 + Math.random() * (100 - 1 * 1));
        list.add(i);
    }
    System.out.println(list);
}

```

2、读取 D 盘上的 English.txt 文件，将里面的小写字母转化为大写字母，输出到 D 盘的 CapEnglish.txt 文件中。

(答案中的 import 部分可以不写)

```

import java.io.BufferedReader;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
public class CapEnglish {
    public static void main(String[] args) {
        File sourceFile = new File("D:/English.txt");
        File targetFile = new File("D:/CapEnglish.txt");
        try{
            BufferedReader bis = new BufferedReader(new
            FileInputStream(sourceFile));
            BufferedOutputStream bos = new BufferedOutputStream(new
            FileOutputStream(targetFile));
            int ch;
            while ((ch = bis.read()) != -1) {
                if (ch>='a'&&ch<='z') ch += ('A' - 'a');
                bos.write(ch);
            }
            bis.close();
            bos.close();
            System.out.println("文件复制完成");
        } catch (IOException e) {
            System.out.println("文件操作发生异常");
        }
    }
}

```

四、综合题（共 3 题，选做其中任意 2 题，共 30 分，每题 15 分，多做不加分，少做扣分）

1、阐述接口和抽象类的区别。编写程序分别使用接口和抽象类来定义运输工具，运输工具包含 `transport()` 方法，并用火车、飞机分别实现 `transport()` 方法。

答：接口和 `abstract` 类的比较如下：

- (1) `abstract` 类和接口都可以有 `abstract` 方法。
- (2) 接口中只可以有常量,不能有变量；而 `abstract` 类中即可以有常量也可以有变量。
- (3) `abstract` 类中也可以有非 `abstract` 方法，接口不可以。

抽象类定义运输工具的程序：

```
abstract class AbsTransport {
    abstract void transport();
}
class AbsTrain extends AbsTransport {
    public void transport() {
        System.out.println("用火车运输!");
    }
}
class AbsPlane extends AbsTransport {
    public void transport() {
        System.out.println("用飞机运输!");
    }
}
public class Test_Abstract {
    public static void main(String args[]) {
        new AbsTrain().transport();
        new AbsPlane().transport();
    }
}
```

接口定义运输工具的程序：

```
interface IntTransport {
    void transport();
}
class IntTrain implements IntTransport {
    public void transport() {
        System.out.println("用火车运输!");
    }
}
class IntPlane implements IntTransport {
    public void transport() {
        System.out.println("用飞机运输!");
    }
}
```

```

public class Test_Interface {
    public static void main(String args[]) {
        new IntTrain().transport();
        new IntPlane().transport();
    }
}

```

2、请说明什么是 Java 中多线程的同步问题和死锁问题，并编写一个多线程程序：实例化 2 个线程，一个线程打印字符 A-T，一个线程打印数字 1-60，2 个线程同时执行，要求打印格式为 A123 B456 T585960 （请写出主要代码）。

答：同步线程, 主要用于一个进程中多个线程的协同工作。线程的同步用于保护线程共享数据，控制和切换线程的执行，保证内存的一致性，Java 提供 synchronized 关键字来实现同步线程。死锁是这样一种情形：多个线程同时被阻塞，它们中的一个或者全部都在等待某个资源被释放。由于线程被无限期地阻塞，因此程序不可能正常终止。

java 死锁产生的四个必要条件：

1. 互斥使用，即当资源被一个线程使用(占有)时，别的线程不能使用
2. 不可抢占，资源请求者不能强制从资源占有者手中夺取资源，资源只能由资源占有者主动释放。
3. 请求和保持，即当资源请求者在请求其他的资源的同时保持对原有资源的占有。
4. 循环等待，即存在一个等待队列：P1 占有 P2 的资源，P2 占有 P3 的资源，P3 占有 P1 的资源。这样就形成了一个等待环路。

当上述四个条件都成立的时候，便形成死锁。当然，死锁的情况下如果打破上述任何一个条件，便可让死锁消失。

解决死锁问题的方法是：一种是用 synchronized，一种是用 Lock 显式锁实现。

代码参考如下

```

public class P401 {
    public static void main(String[] args) {
        Object object = new Object();

        new Thread(new Character(object)).start();
        new Thread(new Number(object)).start();
    }
}

```

```

class Number implements Runnable {

```

```

    private Object object;

```

```

    public Number(Object object) {
        this.object = object;
    }

```

```

    }

    public void run() {
        synchronized (object) {
            for (int i = 1; i <= 60; i++) {
                System.out.print(i);
                if (i > 1 && i % 3 == 0) {
                    System.out.print(" ");
                }
                if (i % 3 == 0) {
                    // 先释放锁,唤醒其他线程,再使本线程阻塞
                    object.notifyAll();

                    if (i < 60) { // 线程执行的最后一次不能堵塞
                        try {
                            object.wait();
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                    }
                }
            }
        }
    }
}

```

```

class Character implements Runnable {

    private Object object;

    public Character(Object object) {
        this.object = object;
    }

    @Override
    public void run() {
        synchronized (object) {
            for (char i = 'A'; i <= 'T'; i++) {
                System.out.print(i);
                // 先释放锁,唤醒其他线程,再使本线程阻塞
                object.notifyAll();
                try {
                    object.wait();
                }
            }
        }
    }
}

```


3、根据下面的 UML，写出程序的主要框架。



```
public class GradeBook {
    private String courseName;
    private Grade[] gradeList;
    private int size;

    public GradeBook() {
    }

    // expand gradeList to 2 times of original length
    private void expand() {
        Grade[] newList = new Grade[gradeList.length * 2];
        System.arraycopy(gradeList, 0, newList, 0,
            gradeList.length /* size is also ok */);
        gradeList = newList;
    }

    public String getCourseName() {
        return courseName;
    }

    public void setCourseName(String courseName) {
        this.courseName = courseName;
    }
}
```

```

    }

    public void findStudentGrade(String stuName) {
        for (int i = 0; i < size; ++i) {
            boolean found = false;
            if (gradeList[i].getStudentName().equals(stuName)) {
                found = true;
                System.out.printf("name: %s, score: %d\n",
                    stuName, gradeList[i].getScore());
            }
            if (!found)
                System.out.printf("Student %s not found!\n",
                    stuName);
        }
    }

    public void insertGrade(Grade g) {
        if (size >= gradeList.length) {
            expand();
        }
        gradeList[size++] = g;
    }
}

class Grade {
    private String studentName;
    private int score;

    public String getStudentName() {
        return studentName;
    }

    public void setStudentName(String studentName) {
        this.studentName = studentName;
    }

    public int getScore() {
        return score;
    }

    public void setScore(int score) {
        this.score = score;
    }
}

```

}

满绩小铺QQ: 1433397577