类和对象作业

一、 填空题

1. 类是组成 Java 程序的基本要素，类体有两部分构成：一部分是变量的定义，另一部分是__方法__的定义。
2. 执行 Person p = new Person();语句后，将在__堆内存__中给 Person 对象分配空间，并在栈内存中给引用变量 p 分配空间，存放 Person 对象的引用。
3. __构造方法__是一种特殊方法，它的名字必须与它所在的类的名字完全相同，并且不书写返回值类型，在创建对象实例时由 new 运算符自动调用。
4. 局部变量的名字与成员变量的名字相同，若想在该方法内使用成员变量，必须使用关键字__this__。
5. 使用关键字__this__来调用同类的其它构造方法，优点同样是以最大限度地代码的利用程度 ，减少程序的维护工作量。
6. 用关键字__static__修饰的成员变量是类变量，类变量是指不管类创建了多少对象，系统仅在第一次调用类的时候为类变量分配内存，所有对象共享该类的类变量。
7. 使用 static 修饰的变量称为静态变量，静态变量可以有两种访问方式，分别是类名.静态变量名和__类名.静态方法名__。
8. 在一个类文件中的关键字 package， import， class 出现的可能顺序是__package, import, class__
9. __package__关键字作为 Java 源文件的第一条非注释性语句，指明该源文件定义的类所有的包。
10. __String__包是 Java 语言的核心类库，它包含了运行 Java 程序必不可少的系统类，使用该包下的类和接口不需要使用 import 导入。

二、 选择题

1. **在 Java 中，以下程序编译运行后的输出结果为（ D ）。（选择一项）**

```
public class Test {
    int x, y;
    Test(int x, int y) {
        this.x = x;
        this.y = y;
    }
    public static void main(String[] args) {
        Test pt1, pt2;
        pt1 = new Test(3, 3);
        pt2 = new Test(4, 4);
        System.out.print(pt1.x + pt2.x);
    }
}
```

A 6

B. 34

C. 8

D. 7

2. 分析如下 Java 程序的代码所示，则编译运行后的输出结果是（ C ）。（选择一项）

```
public class Test {
    int count=9;
    public void count1(){
        count=10;
        System.out.println("count1="+count);
    }
    public void count2(){
        System.out.println("count2="+count);
    }
    public static void main(String[ ] args) {
        Test t=new Test();
        t.count1();
        t.count2();
    }
}
```

A count1=9;
count2=9;

B. count1=10;
count2=9;

C. count1=10;
count2=10;

D. count1=9;
count2=10;

3. 以下语句中关于 Java 构造方法的说法错误的是（ B ）。（选择一项）

A. 构造方法的作用是为创建对象进行初始化工作，比如给成员变量赋值

B. 一个 Java 类可以没有构造方法，也可以提供 1 个或多个构造方法

C. 构造方法与类同名，不能书写返回值类型

D. 构造方法的第一条语句如果是 super()，则可以省略，该语句作用是调用父类无参数的构造方法

4. 在 Java 中关于静态方法，以下说法中正确的是（ AC）。（选择两项）

A 静态方法中不能直接调用非静态方法

B. 非静态方法中不能直接调用静态方法

C. 静态方法可以用类名直接调用

D. 静态方法里可以使用 this

5. 下列选项中关于 Java 中类方法的说法错误的是（ AC ）。（选择二项）

A 在类方法中可用this来调用本类的类方法

B. 在类方法中调用本类的类方法时可直接调用

C. 在类方法中只能调用本类中的类方法

D. 在类方法中调用实例方法需要先创建对象

三、 判断题

1. 类可以看成一类对象的模板，对象可以看成该类的一个具体实例。(T)
2. 如果没有为类中的某些成员赋初始值，Java 会为类成员赋予固定的初始值，如数值变量的值为 0，布尔变量的值为 true，未初始化的引用为 null。（ F ）
3. Java 中所有的变量，不管是成员变量还是局部变量，在使用前都必须进行初始化。（ F ）
4. 在 Java 中对象可以赋值，只要使用赋值运算符即可，相当于生成了一个各属性与赋值对象相同的新对象。（ F ）
5. System.out.println("Hello java!")中 out 是 System 类的一个静态成员变量。（ T ）
6. 构造方法用于创建对象，一般不允许有任何返回值，因此需要在构造方法返回类型处标注为 void。（ F ）
7. 构造方法的作用是两个：一个构造类的对象，另一个作用是初始化对象的属性。(T)
8. Java 语言为所有的 Java 程序自动导入包“java.lang”，因此 Java 程序可以直接用“java.lang”中的类和接口。（ T ）
9. 构造方法的名称必须保持跟类名一致。（ T）

四、 简答题

1. 面向过程和面向对象的区别和联系。
2. 类和对象的关系
3. 局部变量、成员变量、静态变量分别怎么声明?
4. 构造方法的作用和特征
5. this 关键字的作用和用法。
6. 简述 static 关键字的作用。
7. 提示：从 static 可以修饰变量，方法，代码块，内部类四个方面来回答。
8. 栈的特点是?存放什么内容?堆得特点是?存放什么内容?
9. 方法区的特点是?存放什么内容? 方法区和堆有什么共同点?
10. 如果同时导入：import java.util.Date; import java.sql.Date; 我们在程序中怎么区分?

五、 编码题

1. 请定义一个交通工具(Vehicle)的类其中有: 属性：速度(speed)、体积(size)等，方法：移动(move())、设置速度(setSpeed(int speed))、加速 speedUp()、减速 speedDown()等。最后在测试类 Vehicle 中的 main()中实例化一个交通工具对象并通过方法给它初始化 speed,size 的值并且通过打印出来。另外调用加速、减速的方法对速度进行改变。

2. 编写 Java 程序 用于显示人的姓名和年龄。定义一个人类 Person。 该类中应该有两个私有属性： 姓名 （name） 和年龄 （age） 。定义构造方法用来初始化数据成员。再定义显示（display()） 方法将姓名和年龄打印出来。在 main 方法中创建人类的实例然后将信息显示。

1.面向过程和面向对象的区别和联系。
面向过程和面向对象功能都是一样的，面向对象可以增加代码的可读性以及减少重复性。
2. 类和对象的关系
对象是由类创建的
3. 局部变量、成员变量、静态变量分别怎么声明? 成员变量是大家都可以用的，局部变量是只有在方法体内可以被识别的，静态变量是由 static 关键字创建的
4. 构造方法的作用和特征
方法名必须和类名一致。为了可以传入初始值
5.this 关键字的作用和用法。
this 关键字表示对当前类某个对象的引用
简述 static关键字的作用。
提示:从 static可以修饰变量，方法，代码块，内部类四个方面来回答。
变量:只要静态变量
所在的类被加载，这个静态变量就会被分配空间
方法:不需要创建对象就可以执行
代码块:JVM 在加载类时会执行静态代码块
内部类:可以不依赖于外部类实例对象而被实例化
6. 栈的特点是?存放什么内容?堆得特点是?存放什么内容?
栈中存放的是局部变量，变量使用完毕后会从内存中自动释放。堆中存放的都是实体或者是成员变量，存放的实体都是有首地址值的
7. 方法区的特点是?存放什么内容? 方法区和堆有什么共同点?
方法区只是用于存储类,常量相关的信息
8. 如果同时导入:import java.util.Date; import java.sql.Date; 我们在程序中怎么区分?
一个是 util 包另一个是 sql 包

1.

```java
public class Vehicle {
    10 个用法
    private Integer speed;
    4 个用法
    private Double size;
    1 个用法  新 *
    public Vehicle(Integer speed, Double size) {
        this.speed = speed;
        this.size = size;
    }

    1 个用法  新 *
    protected Integer speedUp(){
        this.speed = speed + 1;
        return this.speed;
    }

    1 个用法  新 *
    protected Integer speedDown(){
        this.speed = speed - 1;
        return this.speed;
    }

    0 个用法  新 *
    public void setSpeed(Integer speed) { this.speed = speed; }

    0 个用法  新 *
    public void setSize(Double size) { this.size = size; }

    0 个用法  新 *
    public Integer getSpeed() { return speed; }

    0 个用法  新 *
    public Double getSize() { return size; }
```

```java
新 *
@Override
public String toString() {
    return "Vehicle{" +
            "speed=" + speed +
            ", size=" + size +
            '}';
}

新 *
public static void main(String[] args) {
    Vehicle vehicle = new Vehicle( speed: 120,  size: 100.0);
    System.out.println(vehicle.toString());
    vehicle.speedUp();
    System.out.println(vehicle.toString());
    vehicle.speedDown();
    System.out.println(vehicle.toString());
}
```

2.

```java
public class Person {
    2 个用法
    private String name;
    2 个用法
    private Double age;

    1 个用法  新 *
    public Person(String name, Double age) {
        this.name = name;
        this.age = age;
    }


    1 个用法  新 *
    public String display() {
        return "Person{" +
                "name='" + name + '\'' +
                ", age=" + age +
                '}';
    }

    新 *
    public static void main(String[] args) {
        Person person = new Person( name: "Osmond",  age: 22.5);
        System.out.println(person.display());
    }
}
```

3

```java
public class Circle {
    4 个用法
    private Double r;
    1 个用法
    private Double point;

    1 个用法  新 *
    public Circle(Double r, Double point) {
        this.r = r;
        this.point = point;
    }

    1 个用法  新 *
    protected Double getArea(){
        return Math.PI * r *r;
    }

    1 个用法  新 *
    protected Double getPerimeter(){
        return Math.PI * r * 2;
    }

    新 *
    public static void main(String[] args) {
        Circle circle = new Circle( r: 2.5,  point: 1.0);
        System.out.println(circle.getArea());
        System.out.println(circle.getPerimeter());

    }
}
```

4

```java
public class Point {
    3 个用法
    private Double x;
    3 个用法
    private Double y;

    2 个用法  新 *
    public Point(Double x, Double y) {
        this.x = x;
        this.y = y;
    }

    3 个用法  新 *
    public Double getX() {
        return x;
    }

    0 个用法  新 *
    public void setX(Double x) {
        this.x = x;
    }

    3 个用法  新 *
    public Double getY() {
        return y;
    }

    0 个用法  新 *
    public void setY(Double y) {
        this.y = y;
    }
}
```

```java
1 个用法  新 *
protected Boolean estimatePointInsideCircle(Point p){
    double estimate =  Math.sqrt((p.getX() - this.point.getX()) + (p.getY() - this.point.getY()));
    return estimate <= r;
}

新 *
@Override
public String toString() {
    return "Circle{" +
            "r=" + r +
            ", point= (" + point.getX() + ", " + point.getY() +
            ") }";
}

新 *
public static void main(String[] args) {
    Circle circle = new Circle( r: 2.5, new Point( x: 0.0,  y: 0.0));
    System.out.println(circle.getArea());
    System.out.println(circle.getPerimeter());
    System.out.println(circle.toString());
    System.out.println(circle.estimatePointInsideCircle(new Point( x: 0.1,  y: 0.2)));
}
}
```

5

```java
public class User {
    2 个用法
    private String userId;
    2 个用法
    private String userPassword;
    3 个用法
    private String userEmail;

    2 个用法  新 *
    public User(String userId, String userPassword) {
        this.userId = userId;
        this.userPassword = userPassword;
        this.userEmail = userId.concat( str: "@gameschool.com");
    }

    1 个用法  新 *
    public User(String userId, String userPassword, String userEmail) {
        this(userId, userPassword);
        this.userEmail = userEmail;
    }

    新 *
    @Override
    public String toString() {
        return "User{" +
                "userId='" + userId + '\'' +
                ", userPassword='" + userPassword + '\'' +
                ", userEmail='" + userEmail + '\'' +
                '}';
    }
```

```java
public static void main(String[] args) {
    User user = new User( userId: "junhaocai",  userPassword: "123");
    System.out.println(user.toString());

    User user1 = new User( userId: "cjh", userPassword: "111", userEmail: "123@gmail.com");
    System.out.println(user1.toString());
}
}
```

3. 定义一个圆类——Circle，在类的内部提供 2 个属性：半径(r) 和圆心 Point point，同时提供两个方法：计算面积（ getArea() ）和计算周长（getPerimeter()）。通过两个方法计算圆的周长和面积并且对计算结果进行输出。最后定义一个测试类对 Circle 类进行使用。

4. 为上述 Cricle 类添加一个方法，计算一个点（Point 对象）是否在圆（Cricle 对象）内，并写程序验证。
5. 构造方法与重载：定义一个网络用户类，要处理的信息有用户 ID、用户密码、 email 地址。在建立类的实例时把以上三个信息都作为构造函数的参数输入，其中用户 ID 和用户密码时必须缺省时 email 地址是用户 ID 加上字符串"@gameschool.com"。

六、 可选题

1. 定义一个类 Calculaion， 其中包含四个方法： 加（add()） 、 减（sub()） 、乘（times()） 和除（div()） 。 创建一个具有 main()函数的类。 在 main()函数中创建一个 Calculation 的实例对象并对其中的方法进行调用。
2. 定义一个类 Draw ，在类中提供 3 个方法：输出直角三角形（drawTrian()）、输出矩形（drawRec()） 及平行四边形（drawPra()） 。通过方法可以输出由“*”组成的一个图形。同时在类中包含两个属性：星号的个数（count）、行数（lines）。最后在测试类中进行调用。

3. 创建一个空调，并调用制冷、制热、通风功能。空调包含的属性有品牌、匹数、温度，功能有加热、制冷、通风等功能。

4. 定义一个表示学生信息的类 Student，要求如下：

   （1）类 Student 的属性如下：

   sNO 表示学号； sName 表示姓名； sSex 表示性别； sAge 表示年龄；sJava：表示 Java 课程成绩。

   （2）类 Student 带参数的构造方法：

   在构造方法中通过形参完成对成员变量的赋值操作。

   （3）类 Student 的方法成员：

   getNo（）：获得学号；

   getName（）：获得姓名；

   getSex（）：获得性别；

   getAge（）获得年龄；

   getJava（）：获得 Java 课程成绩

   （4） 根据类 Student 的定义，创建五个该类的对象，输出每个学生的信息，计算并输出这五个学生 Java 语言成绩的平均值，以及计算并输出他们 Java 语言成绩的最大值和最小值。
5. 定义自己的类：手机类（没有标准答案，按照自己的想法定义，一定要包含属性和方法）。
6. 简答题
   1) 使用面向对象的方式分析一个事例(模仿老师的解放战争分析)

2) 局部变量使用之前，必须要手动初始化吗?
3) 如果不手动指定成员变量的值，系统将会自动初始化。那么初始化的规则是?
4) 成员变量从属于谁？静态变量又叫什么以及从属于谁？局部变量从属于谁?
5) 构造方法有没有返回值?详细用文字描述返回值问题。
6) 构造方法如何被调用?
7) 构造方法中能不能有 return 语句?
8) 系统一定会给我们添加无参数的构造方法吗？请详细解释。
9) 下面的代码有什么问题:

```
class Computer {
	int price;
	int type;
	String brand;
	public void start() {
		System.out.println("启动中....");
	}
	Computer(int _price, int type, String _brand) {
		// this.price = price;
		price = _price;
		type = type;
		brand = _brand;
	}
}
```

10) 构造方法能不能重载?
11) 一个构造方法调用另一个构造方法怎么调用? this()，这样的调用方式必须位于第一句吗?
12) static 变量在内存中放置在哪个区? static 变量和方法为什么被称为类变量和类方法?可以被该类所有对象共享吗?
13) 静态初始化块和 main 方法哪个先被执行?
14) 画出如下程序的内存结构：

```
class Engine {
	int speed;
	int weight;
}

class Car {
	static int tyreNum = 4;
	Engine engine;
	String color; // char sequence :字符序列
	void changeColor(String c) {
		color = c;
	}
	void showColor() {
		System.out.println("我的颜色是：" + color);
```

```
    }
}

//测试类和对象
public class TestObject {
    public static void main(String[] args) {
        Car c1 = new Car();
        c1.changeColor("红色");
        c1.showColor();
        System.out.println(Car.tyreNum);
        System.out.println(c1.tyreNum);

        Car c2 = new Car();
        Engine e = new Engine();
        e.speed = 1000;
        e.weight = 10;
        c2.engine = e;
        c2.color = "黑色";

        c2.tyreNum = 10;
        System.out.println(c1.tyreNum);
    }
}
```

15) package 的两个作用是什么?
16) 增加 package 以后，我们在 DOS 下编译怎么做?
17) import 是用于导入包还是导入类?
18) import java.util.*；会不会降低程序运行速度?为什么?
19) import static 静态导入的作用是导入类还是导入类的静态属性和静态方法?
20) javadoc 注释怎么写?
21) java 项目的 API 文档如何生成?请将步骤写出。