厦门大学《Java 程序设计》课程试卷

信息学院　2021级

maoli

一、　单项选择题（共 10 分，每小题 1 分）

1. 以下标识符中，不是 Java 语言关键字的是（　A　）。

A. wait　　B. long

C. new　　D. switch

2. 以下数据类型转换中，必须进行强制类型转换的是（　A　）。

A. int→char　　B. short→long

C. float→double　　D. byte→int

强制类型转换是指将一个数据类型的值转换为另一个数据类型的值，需要使用显式的类型转换操作符。在Java中，将一个整数类型（如int）转换为字符类型（char）时，必须进行强制类型转换。

例如，将int类型的整数赋值给char类型的变量时，需要使用强制类型转换操作符。这是因为int类型是一个整数类型，而char类型表示一个Unicode字符，它们在内部存储方式和表示范围上有所不同。强制类型转换可以确保数据在类型转换过程中的正确性。

其他选项中的类型转换不需要进行强制类型转换：

- short→long：short类型可以自动转换为long类型，因为short的范围在long的范围内。
- float→double：float类型可以自动转换为double类型，因为double类型具有更大的范围和精度。
- byte→int：byte类型可以自动转换为int类型，因为int类型的范围比byte类型大。

3. 下面关于 Java 语言的说法中,错误的是（　B　）。

A. Java 是一个面向对象的语言

B. Java 语言中有指针、结构和类型定义的概念

C. Java 语言是与平台无关的,可以在不同的操作系统下运行

D. Java 语言具有自动无用内存回收机制

在Java语言中，虽然具有引用类型，但它没有显式的指针概念。Java中的引用类型用于引用对象，并且通过引用可以访问对象的成员和方法。相比于指针，Java的引用类型更加安全和简化，不需要像指针一样直接操作内存地址。
Java也不支持结构体（struct）的概念。在C语言中，结构体是一种用户自定义的数据类型，可以包含多个不同类型的数据成员。而在Java中，类是用于创建对象的基本单位，可以封装数据和行为。

4. Java Application 中的主类需包含 main 方法，main 方法的返回类型是（ D ）。

A. int　　B. double

C. float　　D. void

5. 下面（ D ）是 public int method(){…}的重载函数。

A. int method(){…}

B. public int method(){…}

C. public void method(){…}

D. public void method(int m){…}

6. 以下关于构造函数的描述错误的是（ A ）。

A. 构造函数的返回类型只能是 void 型。

B. 构造函数的方法名必须与类名相同。

C. 构造函数的主要作用是完成对类的对象的初始化工作。

D. 一般在创建新对象时，系统会自动调用构造函数。

7. 如果类中的成员变量只可以被同一包访问，则使用如下哪个约束（ D ）。

A. private　　B. protected

C. public　　D. 没有约束符

	同一个类	同一个包	同一个包的子类	不同包的子类	不同包的类
public	可以	可以	可以	可以	可以
protected	可以	可以	可以	可以	不可以
friendly	可以	可以	可以	不可以	不可以
private	可以	不可以	不可以	不可以	不可以

8. 下面哪个流是面向字符的输入（ D ）。

A. FileInputStream　　B. BufferedWriter

C. ObjectInputStream　　D. InputStreamReader

A. FileInputStream是面向字节的输入流，用于读取字节数据。
B. BufferedWriter是面向字符的输出流，用于写入字符数据。
C. ObjectInputStream是面向对象的输入流，用于读取Java对象的串行化表示。
因此，选项D. InputStreamReader是面向字符的输入流，用于读取字符数据。

9. 实现下列哪个接口，可以启用比较功能（ D ）。

A. Runnable　　B. Iterator

C. Serializable　　D. Comparator

10. 在编写访问数据库的 Java 程序时，ResultSet 对象的作用是（ B ）。

A. 用来表示与数据库的连接

B. 存储查询结果

C. 在指定的连接中处理 SQL 语句

D. 建立新数据库连接

二、 判断题（共 10 分，每小题 1 分，正确的打 √,错误的打 X。）

1. Java 源程序文件中是不区分字母的大小写的。X
2. 利用 equals()方法判定 Object 类时，判断的是对象的值是否相等。X（错误）：利用equals()方法判定Object类时，判断的是对象的引用是否相等，而不是值是否相等。Object类中的equals()方法是用来比较两个对象的引用是否指向同一个内存地址。
3. 调用 System.gc()方法可以保证 JVM 立即进行垃圾收集。X（错误）：调用System.gc()方法不能保证JVM立即进行垃圾收集。System.gc()方法只是向JVM建议进行垃圾收集，但具体的垃圾收集时机仍由JVM决定。
4. 调用 sleep()方法可以使一个线程终止运行。X sleep()方法会暂时挂起当前线程的执行，以便给其他线程执行的机会，但线程仍然保持存活状态。
5. 当系统调用当前类的构造方法时，若没有 this()语句进行重载调用，也没有 super()语句调用父类构造方法，则直接执行构造方法中的其他语句。X（错误）：当系统调用当前类的构造方法时，如果没有this()语句进行重载调用，也没有super()语句调用父类构造方法，则会默认调用父类的无参构造方法，而不是直接执行构造方法中的其他语句。
6. 异常处理中，多个 catch 段的参数类之间不能有继承关系。X（错误）：异常处理中，多个catch段的参数类之间可以有继承关系。Java的异常处理机制允许多个catch块按照特定的顺序捕获异常，参数类之间可以有继承关系，但要注意catch块的顺序，遵循从子类到父类的顺序捕获异常。
7. 抽象类可以被直接实例化。X
8. HashMap 以键-值对的方式存储对象。Y
9. Collection 是针对 Collections 集合操作的工具类。X（错误）：Collection是Java集合框架中的一个接口，而不是针对Collections集合操作的工具类。Collections是一个工具类，提供了一系列静态方法，用于对集合进行常见的操作，例如排序、搜索、替换等。
10. Java 程序里,创建新的类对象用关键字 new,回收无用的类对象使用关键字 free。X（错误）：在Java程序中，创建新的类对象使用关键字new，而回收无用的类对象则由Java的垃圾回收器自动处理，无需使用关键字free。Java的垃圾回收机制会自动检测和回收不再被引用的对象，开发人员无需显式地进行对象的回收操作。

三、 简答题（共 30 分，每小题 6 分）

1. 类和接口的区别。

1. 定义：
    - 类（Class）：类是面向对象编程的基本构建块，它是一种具体的数据类型，用于描述对象的属性和行为。类可以包含成员变量、方法、构造函数等，并可以被实例化为对象。
    - 接口（Interface）：接口是一种抽象类型，它定义了一组抽象方法的集合。接口没有具体的实现，只是描述了类应该具有的方法。接口可以被类实现（implements），表示类遵循了接口定义的方法。
2. 继承：
    - 类：类之间可以通过继承关系创建父子关系。子类可以继承父类的属性和方法，并可以添加新的属性和方法或重写父类的方法。
    - 接口：接口之间可以通过继承关系创建扩展关系。一个接口可以继承另一个接口，子接口可以继承父接口的方法定义，并可以添加新的方法。
3. 实例化：
    - 类：类可以被实例化为具体的对象，通过关键字new来创建对象，然后可以使用对象调用类中的属性和方法。
    - 接口：接口本身不能被实例化，但可以被类实现。一个类可以实现一个或多个接口，表示它遵循了接口定义的方法。
4. 多继承：
    - 类：Java中的类是单继承的，一个类只能继承自一个父类。
    - 接口：接口支持多继承，一个类可以实现多个接口，从而获得多个接口定义的方法。
5. 访问修饰符：
    - 类：类的成员变量和方法可以使用不同的访问修饰符进行控制，如public、private、protected和默认（不加修饰符）。
    - 接口：接口的方法默认为public修饰符，成员变量默认为public、static和final修饰符（常量）。
6. 实现与扩展：
    - 类：类通过继承来实现代码的复用和共享，子类可以继承父类的属性和方法。
    - 接口：接口通过实现来实现代码的复用和共享，一个类可以实现多个接口，实现接口定义的方法。

总结：类和接口在Java中具有不同的角色和功能。类用于描述具体对象的属性和行为，并支持继承和实例化。接口用于定义一组抽象方法，描述类应该具有的行为，并支持多继承和实现。类和接口在面向对象编程中发挥着不同的作用，可以根据需求选择适当的使用方式。

2. 举例说明什么是匿名内部类。

匿名内部类是指在使用某个类或接口的实例化对象时，直接定义并创建一个继承该类或实现该接口的子类或实现类的对象，而不需要单独定义一个具名的类。
下面是一个简单的示例，说明匿名内部类的使用：

```
public interface Greeting {
    void sayHello();
}

public class Main {
    public static void main(String[] args) {
        Greeting greeting = new Greeting() {
            @Override
            public void sayHello() {
                System.out.println("Hello, world!");
            }
        };

        greeting.sayHello();
    }
}
```

在上面的例子中，我们定义了一个接口**Greeting**，它只有一个抽象方法**sayHello()**。然后，在**Main**类的**main**方法中，我们直接通过匿名内部类创建了一个实现了**Greeting**接口的对象，并实现了**sayHello()**方法的逻辑。这个匿名内部类没有单独的类名，直接在代码中定义和创建。

3. java 异常处理的五个关键字及其作用。

1. try：try关键字用于定义一个包含可能会抛出异常的代码块。在try块中的代码被监视，一旦发生异常，程序会跳转到对应的catch块进行异常处理。
2. catch：catch关键字用于捕获try块中抛出的异常。catch后面跟着异常类型的参数，用于指定要捕获的异常类型。当try块中发生指定类型的异常时，对应的catch块将被执行，并提供异常处理的逻辑。
3. finally：finally关键字用于定义一个在try-catch块结束后一定会执行的代码块。不管是否发生异常，finally块中的代码都会被执行。通常在finally块中进行一些资源的释放或清理操作。
4. throw：throw关键字用于手动抛出一个异常。可以将异常对象抛出到调用者处进行处理。使用throw关键字可以在程序中主动引发异常，以便进行异常处理。
5. throws：throws关键字用于声明一个方法可能会抛出的异常类型。当一个方法中可能会抛出某种异常，但是不想在方法内部处理该异常时，可以使用throws关键字声明该异常，并由调用该方法的代码块进行异常处理。

4. 正则表达式是什么，有什么作用。

正则表达式是一种用于描述和匹配字符串模式的工具。它是由一系列字符和特殊字符组成的模

式，可以用来进行字符串的搜索、替换、验证等操作。在编程中，正则表达式常被用于文本处理、数据提取、数据验证等任务。

正则表达式的作用包括：

1. 字符串匹配：正则表达式可以用来检查一个字符串是否符合某种模式。通过指定模式，可以从文本中找到与之匹配的部分。
2. 字符串搜索和替换：正则表达式可以在一个文本中搜索与模式匹配的字符串，并进行替换。可以用来查找特定格式的字符串，并将其替换为其他内容。
3. 数据提取：通过使用正则表达式，可以从文本中提取出特定格式的数据，如从一个字符串中提取出所有的电子邮件地址、电话号码等。
4. 数据验证：正则表达式可以用来验证输入数据是否符合特定的规则或格式。例如，可以使用正则表达式验证用户输入的邮件地址、密码强度等。
5. 格式化字符串：正则表达式可以用来格式化字符串，使其符合特定的要求。例如，可以使用正则表达式对日期、时间等进行格式化。

5. 请列举 JavaFX 中的三个容器类，并说明其布局。

答：GridPane网格布局，将子节点按网格形式防止

BorderPane边界布局，将子节点放置在上、下、左、右和中间五个区域。

HBox水平布局，将子节点按水平方向依次排列

四、 填空题（共 50 分）

1. （3 分）面向对象的三大特点分别是继承、多态、封装。
2. （4 分）如果一个类实现了一个接口，那么它需要实现接口的全部 方法 ，否则该类就必须定义为 抽象类
3. （2 分）使用 UDP 协议开发网络程序时，需要使用的两个类分别为 DatagramPacket 和 DatagramSocket
4. （2 分）如果子类想使用父类中的成员，可以通过关键字 super 引用父类中的成员。
5. （2 分）在程序开发中，要想将一个包中的类导入到当前程序中，可以使用 import 关键字。
6. （2 分）在 java 中，符号常量使用关键字 final 进行定义。
7. （2 分）连接数据库时，需要调用 DriverManager 类的 getConnection() 方法创建连接对象。
8. （2 分）在 Java 语言中有定义 int[ ] a = {22,33,44,55,66,77,88},则 a.length = 7
9. （4 分）Java 实现多线程的方法有两种，一种方法是通过创建 Thread 类的子类实现多线程，另一种方法是定义一个实现 Runnable 接口的类。

10. （4 分）请写出以下程序的输出结果 CBA

```
class output
{
    public static void main(String args[])
    {
       StringBuilder s1 = new StringBuilder("ABC");
       StringBuilder s2 = s1.reverse();
       System.out.println(s2);
    }
}
```

11. （4 分）请写出以下程序的输出结果_____

```
import java.util.*;
class Collection_iterators
{
    public static void main(String args[])
    {
        LinkedList list = new LinkedList();
        list.add(new Integer(2));
        list.add(new Integer(8));
        list.add(new Integer(5));
        list.add(new Integer(1));
        Iterator i = list.iterator();
        Collections.reverse(list);
          Collections.sort(list);
         while(i.hasNext())
              System.out.print(i.next() + " ");
     }
 }
```

输出结果为："1 2 5 8"

在给定的代码中，首先创建了一个**LinkedList**对象**list**，并向其中添加了四个整数对象。然后，通过调用**list.iterator()**方法获取一个迭代器对象**i**，用于遍历集合中的元素。

接下来，通过调用**Collections.reverse(list)**方法对**list**进行反转操作，使集合中的元素顺序颠倒。然后，通过调用**Collections.sort(list)**方法对**list**进行排序，按升序重新排列集合中的元素。

在循环中，使用**i.hasNext()**方法检查是否还有下一个元素，然后通过**i.next()**方法获取下一个元素，并使用**System.out.print()**方法将元素输出到控制台。因为在之前已经对集合进行了排序，所以输出的元素将按照升序排列，最终输出结果为："1 2 5 8"。

12. （9 分）请写一个泛型方法，比较两个参数的大小，若大于，返回一个正数；若等于返回 0；若小于返回一个负数。

```java
public class CompareUtils {
    public static <T extends Comparable<T>> int compare(T obj1, T obj2) {
        return obj1.compareTo(obj2);
    }

    public static void main(String[] args) {
        int result1 = compare(10, 5); // 比较整数
        System.out.println(result1); // 输出正数

        int result2 = compare("abc", "def"); // 比较字符串
        System.out.println(result2); // 输出负数

        int result3 = compare(7.5, 7.5); // 比较浮点数
        System.out.println(result3); // 输出0
    }
}
```

13.（10 分）请完善 removeDuplicate 方法，去除一个 ArrayList 类对象中的重复值，并返回去重后的 ArrayList 对象。

```java
public ArrayList<String> removeDuplicate(ArrayList<String> collection ){

}
```

```java
import java.util.ArrayList;
import java.util.HashSet;

public class RemoveDuplicateExample {
    public static ArrayList<String> removeDuplicate(ArrayList<String> collection) {
        HashSet<String> set = new HashSet<>(collection); // 将 ArrayList 转换为 Hash
        ArrayList<String> result = new ArrayList<>(set); // 将去重后的元素重新构建
        return result;
    }

    public static void main(String[] args) {
        ArrayList<String> collection = new ArrayList<>();
        collection.add("apple");
        collection.add("banana");
        collection.add("apple");
        collection.add("orange");
        collection.add("banana");

        ArrayList<String> uniqueList = removeDuplicate(collection);
        System.out.println(uniqueList);
    }
}
```