

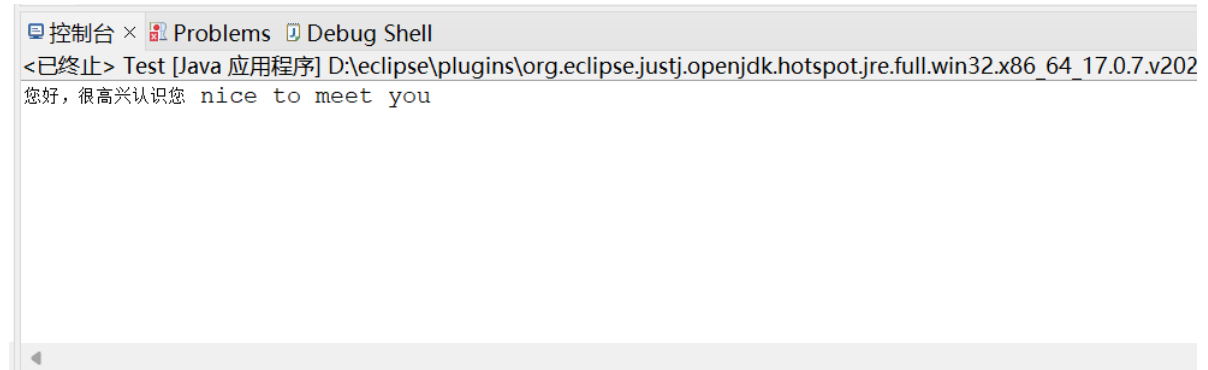
Lab 02 实验报告

班级：222111

学号: 22371001

姓名：卜欣然

Question 1



(1) 源文件的名字: Person.java

(2) 编译上述源文件将生成几个字节码文件:2个。

这些字节码文件的名字都是什么:Person.class, Test.class

(3) 在命令行执行 `java Person` 得到怎样的错误提示:

错误：在类 `lab02.Person` 中找不到 `main` 方法，请将 `main` 方法定义为：`public static void main(String[] args)`

否则 JavaFX 应用程序类必须扩展 `javafx.application.Application`

执行 `java test` 得到怎样的错误提示:

错误：找不到或无法加载主类 lab02.test原因：java.lang.NoClassDefFoundError: lab02/test (wrong name: lab02/Test)

执行 `java Test.class` 得到怎样的错误提示:

错误：找不到或无法加载主类 lab02.Test.class原因：java.lang.ClassNotFoundException: lab02.Test.class

执行 `java Test` 得到怎样的输出结果:

您好，很高兴认识您 nice to meet you

Question 2

```
Problems @ Javadoc 声明 控制台 ×
<已终止> Unicode [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425
'大'的 Unicode 编码: 22823
Unicode 编码为23398的字符是: 学
```

Question 3

```
Problems @ Javadoc 声明 控制台 ×
<已终止> UseIntArray [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425
intArray[0]=8
intArray[1]=9
intArray[2]=12
intArray[3]=13
intArray[4]=14

sum = 56
```

Question 4

```
Problems @ Javadoc 声明 控制台 ×
<已终止> TwoDimensionArray [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425
b[0][0] = 1000
sum = 1139
b.length = 3
arr1:
0 1 2 3 4 5 6 7 8 9 10 11
arr2:
12
13
14
15
16
17
18
19
20
21
22
23
arr3:
0 1 2 3 4 5 6 7 8
```

Question 5

```
Problems @ Javadoc 声明 控制台 ×
<已终止> SwitchExample [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425
Jeep好好
```

原因：

第一遍循环*i*=1时，进入case 1，将*c*赋值为'j'，输出"j"，但由于未遇到break，将会不判断条件，直接继续执行case 2，将*c*赋值为'e'，并输出"e"。接着遇到break后跳出switch语句。

进入第二遍循环，此时*i*=2，进入case2，继续将*c*赋值为'e'，还输出"e"，到break后跳出switch语句。

进入第三遍循环，此时*i*=3，进入case 3，将*c*赋值为'p'并输出"p"，因为未遇到break，将继续执行default中的语句，输出"好"。

进入到第四遍循环，此时*i*=4，进入default，再次输出一个"好"。

i++后*i*=5，此时*i*不符合小于等于4的条件，跳出循环，运行结束。

Question 6

```

<已终止> phalanx [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-150
7
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31 32 33 34 35
36 37 38 39 40 41 42
43 44 45 46 47 48 49

控制台 × Problems Debug Shell
<已终止> phalanx [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_1
5
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25

<已终止> phalanx [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-150\jre\bin\javaw.exe (2023年9月12
10
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60
61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90
91 92 93 94 95 96 97 98 99 100
```

Question 7

★
★ ★ ★
★ ★ ★ ★ ★
★ ★ ★
★

11

Question 10

-45
12
45
67
67
89
123
0
No

```
-45
12
45
67
67
89
123
123
Yes
```

Question 11

控制台 × Problems Debug Shell

```
<已终止> ForInString [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.op  
A n   A f r i c a n   S w a l l o w
```

Question 12

(1) 指出代码中注释标注出的四行输出的内容会是什么：

- 1: p1 的 x, y 坐标: 1111, 2222
- 2: p2 的 x, y 坐标: -100, -200
- 3: p1 的 x, y 坐标: 0, 0
- 4: p2 的 x, y 坐标: 0, 0

(2) 什么是浅拷贝？什么是深拷贝？

浅拷贝：被复制对象的所有变量都含有与原来的对象相同的值，而所有的对其他对象的引用仍然指向原来的对象。即对象的浅拷贝会对“主”对象进行拷贝，但不会复制主对象里面的对象。“里面的对象”会在原来的对象和它的副本之间共享。简而言之，浅拷贝仅仅复制所考虑的对象，而不复制它所引用的对象

深拷贝：是一种**完全拷贝**，无论是值类型还是引用类型都会完完全全的拷贝一份，在内存中生成一个新的对象，简单点说就是拷贝对象和被拷贝对象没有任何关系，互不影响。

(3) 如果你要为一个类实现 `copy()` 方法，思考如何避免引用间赋值导致的浅拷贝？或者说，如何确保进行的是深拷贝。

1.通过构造函数实现深拷贝，直接new一个对象。比如：

```
User copyUser = ``new` `User(user.getName(), ``new` `Address(address.getCity(),  
address.getCountry()));
```

2.重载clone()方法，将它protected类型改写为public类型。

3.Apache Commons Lang序列化。Java提供了序列化的能力，我们可以先将源对象进行序列化，再反序列化生成拷贝对象。但是，使用序列化的前提是拷贝的类（包括其成员变量）需要实现Serializable接口。Apache Commons Lang包对Java序列化进行了封装，我们可以直接使用它。

4.Gson序列化。Gson可以将对象序列化成JSON，也可以将JSON反序列化成对象，所以我们可以用它进行深拷贝。

5.Jackson序列化。

(4) 解释String的 `==` 和 `equals()` 的区别。

`==`相等判断符用于判断基本数据类型和引用数据类型。判断基本数据类型时，判断的是数值，当判断引用数据类型时，判断变量是否指向同一引用对象。

`equals`判断，该方法将逐个地比较两个字符串的每个字符是否相同，不比较是否指向同一引用对象。

Question 13

输出结果：

```
数组 a 的元素个数 = 4
数组 b 的元素个数 = 3
数组 a 的引用 = [I@2133c8f8
数组 b 的引用 = [I@43a25848
数组 a 的元素个数 = 3
数组 b 的元素个数 = 3
a[0] = 100, a[1] = 200, a[2] = 300
b[0] = 100, b[1] = 200, b[2] = 300
```

原因：

初始时，a和b对应的内存区域储存的是不同的地址，分别是[I@2133c8f8和[I@43a25848，这两个地址分别指向堆区中{ 1, 2, 3, 4 }和{ 100, 200, 300 }。当执行a=b时，将b的地址直接赋给a，两个数组对应的内存区域储存的地址均为[I@43a25848。所以a的元素个数为3，调用a[0],a[1],a[2]实质就是调用b[0],b[1],b[2]，所以相同。

Question 14

```
控制台 ^ Problems Debug Shell
<已终止> VarArg [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64
1
2
-1
-2
-3
-4
9
7
6
```

Question 15

```

1 package lab02;
2 public class strstr {
3     public static void main(String [] args) {
4         String output=strscat("a","b","c","", "e");
5         System.out.println(output);
6     }
7     /**
8      * 将任意个字符串顺序连接，不应该改变任意一个原有参数
9      * @param args 字符串们
10     * @return args中的字符串顺序连接组成的新字符串
11     */
12     public static String strscat(String... args){
13         for(int i=1;i<args.length;i++) {
14             args[0]=args[0]+args[i];
15         }
16         return args[0];
17     }
18 }
19
20

```

控制台 × Problems Debug Shell

<已终止> strstr [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\abce

```

sort.java  strstr.java  P1.java
1 package lab02;
2 public class strstr {
3     public static void main(String [] args) {
4         String output=strscat("a","b","c","", "e");
5         System.out.println(output);
6         output=strscat("str");
7         System.out.println(output);
8     }
9     /**
10     * 将任意个字符串顺序连接，不应该改变任意一个原有参数
11     * @param args 字符串们
12     * @return args中的字符串顺序连接组成的新字符串
13     */
14     public static String strscat(String... args){
15         for(int i=1;i<args.length;i++) {
16             args[0]=args[0]+args[i];
17         }
18         return args[0];
19     }
20 }
21
22

```

控制台 × Problems Debug Shell

<已终止> strstr [Java 应用程序] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jr
abce
str

(1) 尝试理解 Java 可变参数的实现机制，说说你的想法。

形参接受多个相同类型的参数值，多个参数值被当成数组传入。所以，通过数组下标即可访问各个参数值。

(2) 调用 `strscat(new String[]{"a", "b"})` 能通过编译吗？为什么？

能通过编译。通过可变参数和数组的形式，这两种调用形式本质上是一样的。

(3) 如果还有静态方法 `String strscat(String[] args)` 同时存在，代码能通过编译吗？给出 IDE 的编译结果的截图。

不能通过编译。可变参数在编译为字节码后，在方法签名中会以数组形态出现的，导致这两个方法的签名一致的，如果同时出现，是不能编译通过的。

错误 (9 项)					
Duplicate method strscat(String...) in type strscr	strscr.java	/lab02/src/lab02	第 16 行	Java 问题	
Duplicate method strscat(String[]) in type strscr	strscr.java	/lab02/src/lab02	第 22 行	Java 问题	

(4) 如果我们声明的是 `String strscat(String[] args)`，`strscat("a", "b", "c")` 这样的调用还能通过编译吗？给出 IDE 的编译结果的截图。

不能通过编译。

