 姓名：李全欣 班级：计算机2018级4-7 成绩：83分**一.单选题** (共1题,8.3分)

1 以下能够正确描述, C:/a/b, 路径的输出是

```
String p1 = "C:/";
```

```
String p2 = "a";
```

```
String p3 = "b";
```

A、 System.out.println(p1 + p2 + p3);

B、 System.out.println(Path.of(p1 + p2 + p3));

C、 System.out.println(Path.of(p1).resolve(p2).resolve(p3));

D、 System.out.println(Path.of(p3).resolve(p2).resolve(p1));

正确答案： C 我的答案： C

得分： 8.3分

二.判断题 (共10题,83.0分)

1 Java中的资源对象与创建的普通对象不同, 不会在失去引用后自动销毁

我的答案：√ 正确答案：√

得分： 8.3分

2 在try语句内创建的任何对象, 在try块结束后直接销毁

我的答案：√ 正确答案：×

得分： 0.0分

3 如果在try-with-resources语句的try代码块内引发异常, 先执行catch块处理异常以及finally块后, 关闭资源

我的答案：×

正确答案：×

得分： 8.3分

4 InputStream的read(byte[] b)方法, 当流中字节数多于字节数组缓冲区容量时, 将抛出异常

我的答案：×

正确答案：×

得分： 8.3分

5 InputStream的read(byte[] b)方法, 当流中没有字节时返回0

我的答案：×

正确答案：×

得分： 8.3分

6 OutputStream的write(byte[] b)方法, 会将字节数组缓冲区中的全部字节写入输出流

我的答案：√ 正确答案：√

得分： 8.3分

7 FileOutputStream是OutputStream接口的一个实现类

我的答案: ×

正确答案: ×

得分: 8.3分

8 Java的Path是用以描述路径的抽象类

我的答案: ×

正确答案: ×

得分: 8.3分

9 Path仅能用于描述路径，无法描述文件名称

我的答案: ×

正确答案: ×

得分: 8.3分

10 Path不依赖于实际运行的操作系统，屏蔽了不同操作系统路径描述符的差异

我的答案: ✓

正确答案: ✓

得分: 8.3分

三.简答题 (共1题,8.7分)

1 ! 过滤代码格式后，粘贴提交！

编写一个方法，传入输入源文件路径，以及输出源文件路径。路径为包含了文件路径/名称/扩展名，的字符串描述

可以是文本文件或任何类型文件

方法实现：基于基本字节数组缓冲区的IO流操作，基于指定输入源/输出源复制文件；资源必须被正确关闭；异常

在方法内处理；无需考虑目录不存在问题

正确答案：

```
private static void copy(String source, String target) {
    try(FileInputStream in = new FileInputStream(source);
        FileOutputStream out = new FileOutputStream(target)) {
        byte[] buffer = new byte[512];
        int len;
        while ((len = in.read(buffer)) != -1) {
            out.write(buffer, 0, len);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

我的答案：

```
import java.nio.file.Files;
import java.nio.file.Path;
```

```
import com.sun.org.apache.xerces.internal.impl.io.UTF8Reader;
import com.sun.xml.internal.stream.writers.UTF8OutputStreamWriter;
public class Main {

    public static void main(String[] args) throws IOException {
        getStringsInText();
        readString();
        copyByInputStream();
    }
    private static void getStringsInText() throws IOException {
        try (FileInputStream in = new FileInputStream("D:/test/cn.txt");
            FileOutputStream out = new FileOutputStream("D:/test/cout.txt");
        ) {
            byte[] buffer = new byte[100];
            int len;
```

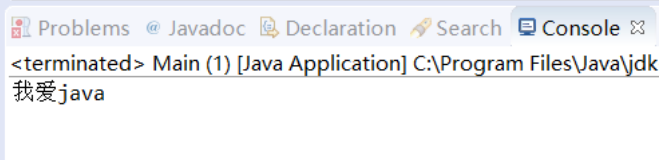
```
while ((len = in.read(buffer)) != -1) {
    out.write(buffer, 0, len);
}




}

private static void readString() throws IOException {
    String result = Files.readString(Path.of("D:/test/cn.txt"));
    System.out.println(result);
}

private static void copyByInputStream() throws IOException {
    InputStream in = new FileInputStream("D:/test/cn.txt");
    Files.copy(in, Path.of("D:/test/cn1.txt"));
}

}
```



	cn.txt	2020/5/14 14:45	文本文档	1 KB
	cn1.txt	2020/5/14 16:28	文本文档	1 KB
	cout.txt	2020/5/14 16:28	文本文档	1 KB

