作业 2 说明（2021.03.26）

1. 要求实现一个自己的字串类，命名为 MyString，类的代码不得使用 java.lang 的 String 类
2. MyString 的主要成员变量为 **final** char [] value 用于存字串
3. MyString 的构造方法
   MyString(char [] v) { //将 v 的内容复制到成员 value
   ...
   }
4. MyString 提供 **public 方法 int indexOf（char [] v）**，如果 v 是 value 的子串，返回第一个元素的 index。否则返回-1.
   例子：对于 MyString a, 如果 a 的成员变量 value 为{'1', '2', '3'}，如果 v1 为{'2', '3'}，v2 为{'1'}, v2 为{'4'}，则 a.indexOf(v1)返回 1，a.indexOf(v2)返回 0，a.indexOf(v3)返回-1.
5. MyString 提供 **public 方法 MyString concat（char [] v）**实现字符串的相连。
   例子：对于 MyString a, 如果 a 的成员变量 value 为{'1', '2', '3'}，如果 v 为{'4'}，则 a.Concat(v)返回一个新的 MyString 类型的对象，其 value 为{'1', '2', '3', '4'}
6. MyString 提供 **public 方法 MyString replace（char [] v1, char [] v2）**实现子串的替换，并返回一个新的 MyString 类型的对象。如果 value 存在和 v1 一样的子串，将之替换为 v2。如果不存在这样的子串，则 value 保持相同的值。

例 1：对于 MyString a, 如果 a 的成员变量 value 为{'1', '2', '2'}，如果 v1 为{'1', '2'}，v2 为{'1'}，则 a.replace (v1, v2)返回一个 MyString 类型的对象，其 value 为{'1', '2'}。

例 2：对于 MyString a, 如果 a 的成员变量 value 为{'1', '2', '1', '2'}，如果 v1 为{'1', '2'}，v2 为{'1'}，则 a.replace (v1, v2)返回一个 MyString 类型的对象，其 value 为{'1', '1'}。

从例子可以看出，替代操作只替代原有的 value 里的子串，不循环重新替代新生成的串里的子串，如例 1，先找到第一个'1', '2'，替换为'1'后，不再认为这个新的'1'与后面的'2'构成的新的'1' '2'而需要被替代。因此结果是{'1', '2'}。

7. MyString 提供 public 方法 **int length()**，得到字串长度
8. MyString 提供 public 方法 **char [] getValue()**，得到字串数组（返回一个新的数组，而不是直接返回内部的值）

9. 效率比较（MyStringTest.java）
   简单的效率测试代码范例

```
String stringOrig =
   "12345678123456781234567812345678123456781234567812345678123456
   812345679";
char [] chars = stringOrig.toCharArray();
String stringToBeFound = "123456789";
char [] dest = stringToBeFound.toCharArray();

String stringToBeReplaced = "1234567";
char [] toBeReplaced = stringToBeReplaced.toCharArray();
```

```
long startTime;
long endTime;

MyString a = new MyString(chars);
startTime = System.currentTimeMillis(); //获取当前时间
for(int i = 0; i < TIMES1; i++) {
   a.indexOf(dest);
   //System.out.println("ret = " + a.indexOf(dest));
}
endTime = System.currentTimeMillis();
System.out.println("Time: "+(endTime-startTime)+"ms");

startTime = System.currentTimeMillis(); //获取当前时间
for(int i = 0; i < TIMES1; i++) {
   stringOrig.indexOf(stringToBeFound);
}
endTime = System.currentTimeMillis();
System.out.println("Time: "+(endTime-startTime)+"ms");


startTime = System.currentTimeMillis(); //获取当前时间
for(int i = 0; i < TIMES2; i++) {
   a.replace(toBeReplaced, dest);
   //System.out.println("ret = " + a.indexOf(dest));
}
endTime = System.currentTimeMillis();
System.out.println("Time: "+(endTime-startTime)+"ms");

startTime = System.currentTimeMillis(); //获取当前时间
for(int i = 0; i < TIMES2; i++) {
   stringOrig.replace(stringToBeReplaced, stringToBeFound);
}
endTime = System.currentTimeMillis();
System.out.println("Time: "+(endTime-startTime)+"ms");
```

10. 在 MyStringTest.java 源代码的结尾，以注释形式加上性能测试结果的输出。
11. 提交的压缩包请务必按照下面的包结构整理，如果**不符合要求，酌情扣分**。

评分考虑:

1. 以上所有作业要求如果有疑问，请以 Java 自带的 String 类的行为为准。如果仍然有疑问，请询问老师和助教。
2. 代码的整洁可读性（注释、编码习惯等，可以考虑英文注释）。
3. 是否能够通过测试用例的测试 （测试用例不会包括输入空引用，如 concat(null)或 replace(null, ..), 但是可能会涵盖其他任何可能的情况。
4. 提交的时间先后不会作为评分的标准。
5. 测试的版本以最后提交的版本为准。

包结构:

提交的压缩包，请务必按照以下的格式整理。

```
.
└── XXX.zip （姓名.zip）
    ├── MyString.java
    ├── MyStringTest.java
    ├── report.pdf （一页A4纸之内，可以有图表）
```

P.S.:

1. 其格式可能与某些 IDE 不同，但请都整理成以上结构（包括 package）。
2. 部分同学可能使用某些单元测试工具（junit 等），但请在提交时去掉单元测试文件，保证结构与以上相同。
3. 不要提交编译后的.class 文件，测试时会自行编译。