Lab2 字符串、数组和文件I/O

- 字符串
- 数组
- 文件I/O

刘诺纬

10191900446@stu.ecnu.edu.cn

字符串

创建字符串

声明和赋值

```
String str;
str = "hello";
```

或者，将声明与赋值在同一行代码内完成

```
String str = "hello";
```

创建字符串的方式类似于基本数据类型

```
int a = 10;
double b = 3.14;
```

但是需要注意的是，String不是基础数据类型（primitive type）！

字符串

字符串的操作

1. `length()`：获取字符串长度

```
String str = "hello";
System.out.println(str.length());  // 5
```

中文字符串同理

```
String str = "你好";
System.out.println(str.length());  // 2
```

字符串

字符串的操作

2. `concat()`：拼接字符串

```
String str1 = "hello";
String str2 = ", world";
System.out.println(str1.concat(str2));  // hello, world
```

除了`concat()`之外，Java还支持`+`操作符拼接字符串

```
System.out.println(str1 + str2);  // hello, world
```

区别：`concat()`的两侧都必须是`String`，不能是`null`或其他类型

```
int a = 10;
System.out.println(str1.concat(a));  // 编译错误
System.out.println(str1 + a);  // hello10
```

字符串

字符串的操作

3. `equals()`：字符串比较

```
String str1 = "hello";
String str2 = "hello";
String str3 = "world";
System.out.println(str1.equals(str2));  // true
System.out.println(str1.equals(str3));  // false
```

`equals()`用于比较字符串的内容，而不是引用（`str1`，`str2`是否指向同一个字符串） 可以使用`==`判断引用

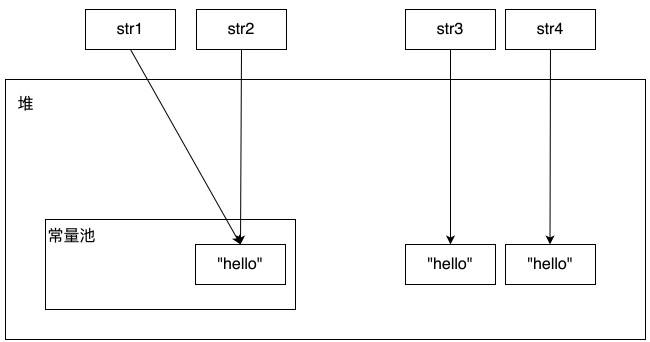
```
System.out.println(str1 == str2);  // true
```

字符串

使用`new`关键字创建字符串

```
String str1 = "hello";
String str2 = "hello";
String str3 = new String("hello");
String str4 = new String("hello");
```

与之前的方法有什么不同?



直接赋值创建的字符串存储在常量池中，而构造函数创建的字符串对象则存储在堆中。

对于内容相同的字符串，在常量池中永远只有一份，在堆中有多份。

字符串

使用`new`关键字创建字符串

```
String str1 = "hello";
String str2 = "hello";
String str3 = new String("hello");
String str4 = new String("hello");

System.out.println(str1 == str2);  // true
System.out.println(str3 == str4);  // false
System.out.println(str1.equals(str2));  // true
System.out.println(str3.equals(str4));  // true
System.out.println(str1.equals(str3));  // true
```

数组

一维数组

声明数组

```
int[] arr1;  // 推荐
int arr2[];
```

初始化数组

```
int[] arr1 = new int[10];
int[] arr2 = {1, 2, 3, 4, 5};
```

与C语言类似的是，数组被创建后，其长度不能被修改

与C语言不同的是，数组初始化存在默认值（例如上面的例子中，`arr1`的每个元素默认都是0）

数组

二维数组

声明数组

```
int[][] arr1;  // 推荐
int arr2[][];
```

初始化数组

```
int[][] arr1 = new int[2][3];

int[][] arr2 = new int[2][];
arr2[0] = new int[3];
arr2[1] = new int[2];

int[][] arr3 = {{1, 2, 3}, {4, 5, 6}};
```

数组

Java ArrayList

```java
List<String> arr = new ArrayList<>();
arr.add("first");
arr.add("second");
arr.add("third");

for (int i=0; i<arr.size(); ++i) {
    System.out.println(arr.get(i));
}

for (String str : arr) {
    System.out.println(str);
}

arr.remove("second");
System.out.println(arr);
```

Python list

```python
arr = []
arr.append("first")
arr.append("second")
arr.append("third")

for i in range(len(arr)):
    print(arr[i])


for str in arr:
    print(str)


arr.remove("second")
print(arr)
```

`ArrayList`是Java中的动态数组

使用前需要`import java.util.List`和`import java.util.ArrayList`

文件I/O

流

- 流（Stream）：一组有顺序的、有起点和终点的字节集合，是对数据传输的总称。也就是说，数据在两个对象之间的传输称为流。
- 按照流的传输方法，可以分为**输入流**和**输出流**。

程序 -> 文件 输出

文件 -> 程序 输入

- 流的基本操作有读操作和写操作，从流中取得数据的操作称为读操作，向流中添加数据的操作称为写操作。对输入流只能进行读操作，对输出流只能进行写操作

文件I/O

FileIO

方法	说明
`void writeStringToFile(String str, String fileName)`	将一个字符串追加地写入指定文件（支持中文）
`char getCharFromFile(int pos, String fileName)`	将读取指定文件的第pos个字符（支持中文）
`String getLineFromFile(int pos, String fileName)`	将读取指定文件的第pos行的字符串（支持中文）
`String[] getAllLinesFromFile(String fileName)`	将读取指定文件的所有行的字符串,并按照顺序储存在String类型数组中返回（支持中文）

附录

字符串常用方法

方法	描述
`int length()`	返回字符串的长度
`String toUpperCase()`	将串中字符变成大写
`String toLowerCase()`	将串中字符变成小写
`char charAt(int index)`	返回位置index处的字符
`String substring(int s, int e)`	返回从位置s到e的字符子串[s,e)
`String substring(int s)`	返回从位置s到末尾的字符子串
`int indexOf(String s)`	返回首次出现字符串s的位置
`int indexOf(String s,int i)`	返回在位置i之后首次出现s的位置
`String trim()`	返回一个新串，去除前后空白字符