Instituto Superior de Engenharia de Coimbra

Advanced Programming (2019/20)

Exercises

`String`

The String class represents character strings. All literal strings, such as "abc", are implemented as instances of this class.

The objects of type String are constant, their values can not be changed after being created.

```
String str = "abc";
```

is the string created below:

```
char data [] = { 'a', 'b', 'c'};
String str = new String (data);
```

Examples:

```
String str = new String(); // empty String
```

or

```
String str = ""; // empty String.

str += "My String";
String anotherStr = str; // anotherStr is a reference to "My String"
str += "aaa"; // str is now a reference to "My Stringaaa"
```

`str` is now a reference to a new String object `"My Stringaaa"`. Concatenation abandons the old object and creates a new object (in this case, `"My Stringaaa"`) with the new string. `anotherString` refers `"My String"`.

Some methods:

`charAt public char(int index)` returns the character corresponding to the position index

`public int compareTo (String anotherString)` compares two strings lexicographically; returns a negative integer, zero or positive as the string represented by the object is less than, equal or greater than the string represented by the argument.

`public int compareToIgnoreCase (String str)` compares two strings lexicographically, it is not case sensitive.

`public boolean equals (Object anObject)` compares the strings represented by the object and the argument.

`public boolean equalsIgnoreCase (String anotherString)` compares the strings and the argument object, ignoring differences between uppercase and lowercase.

`public int indexOf (int ch)` returns the index corresponding to the first occurrence of the character `ch` or `-1` if the string does not contain the character.

`public int length ()` returns the number of characters in the string.

`toLowerCase public String ()` returns the string converted in lowercase.

`toUpperCase public String ()` returns the string converted to uppercase.

StringBuffer and StringBuilder

The StringBuffer and StringBuilder classes represent strings that can be altered (length and contents can be changed). The difference between these two classes is that the methods of StringBuffer class are synchronized (appropriate to muilti-thread use) and methods of StringBuilder are not (being appropriate if the string is accessed only by a thread since its implementation is faster).

```
StringBuffer strB = new StringBuffer ( " my string ");
String buffer outraStrb = strB; // outraStrb is a reference to "my string "
strb.append ( "that changes"); // outraStrb and strB refer
                                // "my string that changes"
```

Some methods:

`public StringBuffer append (string str)` adds the string `str`.

`charAt public char (int index)` returns the character corresponding to the `index` position.

`public int indexOf (String str)` if the string `str` is contained in the string represented by the object returns the index of the first character, otherwise it returns -1.

`public int length ()` returns the number of characters in the string represented.

`public void setCharAt (int index, char ch)` the character in the `index` position is replaceded by `ch`.

Exercises

1. Write an application that implements the hangman game.
2. Define a class to represent the following information about a document:
   - Title
   - Group of authors (versions array of objects and ArrayList of objects)
   - Text

   The class must have the following features:
   - Add an author to the document
   - Removing an author
   - Add text
   - Count the words of text (words can be separated by blank charaters, commas or periods.
   - Replace with uppercase first letters of the words after periods.
   - Count the occurrences of a given word.