Technische Hochschule Rosenheim

Exercise 10 – Characters and strings

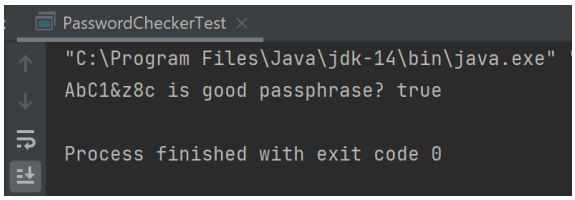
Task 1 – Implement and test the PasswordChecker class

Passwords must meet certain quality criteria. For example, a good password should contain at least

- eight characters,
- two numbers,
- one upper case and one lower case letter, and
- one special character.
  (A special character is a character that is not a number or a Unicode letter.)

a) Create a `PasswordChecker` class. Then write a static method `isGoodPassword` for the class, which checks a passed string that represents the password, based on the above-mentioned criteria. Based on this check, the method should return the result `true` or `false`.

b) Write a suitable `PasswordCheckerTest` test class, which checks the string "`AbC1&z8c`" to see if it is a good password, and then returns the result to the console.



Task 2 – Implement and test the Palindrome class

A **palindrome** is a word or sentence that - ignoring special characters / upper case / lower case -

is identical when read backwards and forwards, such as:

- "Otto"
- "Risotto, Sir?"
- "Do geese see God?"

a) Create a `Palindrome` class. Next, write a static method `filter` for the class, which eliminates all non-letters in a specified string, converts all other characters to lower case, and then returns the generated string as the result.
Think about: what is the method signature for `filter`?

b) Write another static method `isPalindrome` for the `Palindrome` class, which checks whether the string passed is a palindrome. If so, the method returns "`true`".
Think about: what is the method signature for `isPalindrome`?
Think about: what is the best way to reverse the contents of an array?

c) Write a `PalindromeTest` class, which appropriately tests the `Palindrome` class using the methods implemented in (a) and (b). A test sequence could look like this:

Implementation notes:

- You can use all methods of the java.lang.Character class and all methods of the java.lang.String class for implementation.