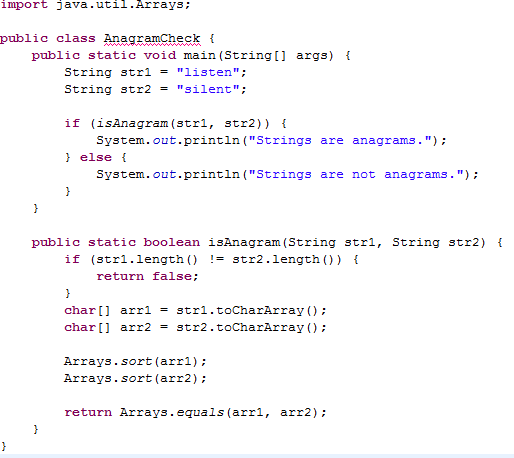
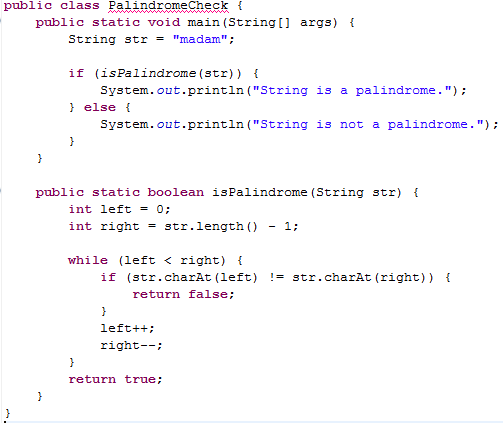
Check if two strings are anagrams



Explanation:

* **if (str1.length() != str2.length())**: Checks if both strings have the same length.
* **Arrays.sort(arr1)**: Sorts the character array of the first string.
* **Arrays.equals(arr1, arr2)**: Compares the sorted character arrays of both strings.

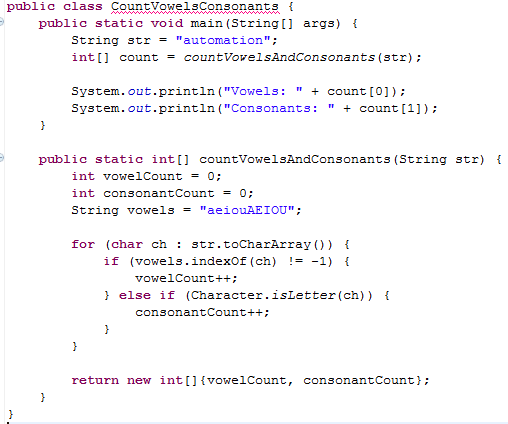
Check if a string is a palindrome



Explanation:

* **while (left < right)**: Loops through the string comparing characters from the start and end.
* **str.charAt(left) != str.charAt(right)**: If characters don't match, it's not a palindrome.

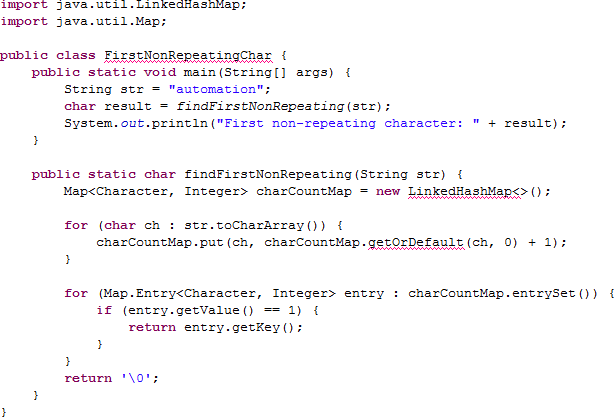
Count the number of vowels and consonants in a string



Explanation:

* **vowels.indexOf(ch) != -1**: Checks if the character is a vowel.
* **Character.isLetter(ch)**: Ensures that only letters are counted as consonants.

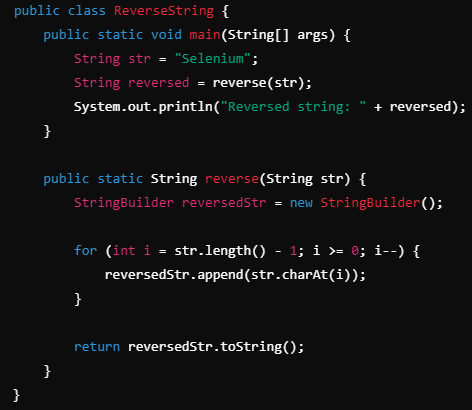
Find the first non-repeating character in a string



Explanation:

* **charCountMap.getOrDefault(ch, 0) + 1**: Increments the count of each character.
* **if (entry.getValue() == 1)**: Finds the first character that appears only once.

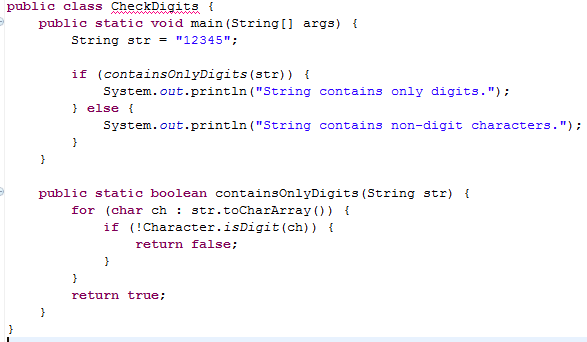
Reverse a string



Explanation:

* + **for (int i = str.length() - 1; i >= 0; i--)**: Loops through the string from the end to the beginning.
  + **reversedStr.append(str.charAt(i))**: Appends each character to the reversed string.

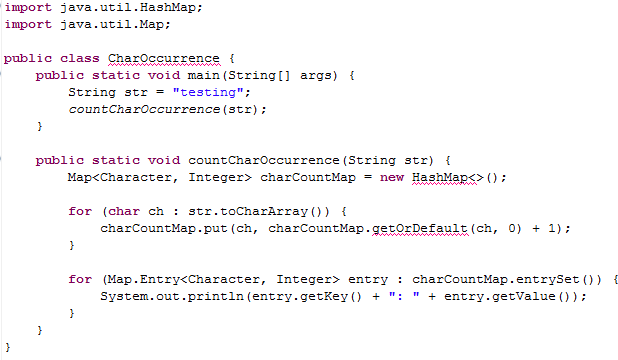
Check if a string contains only digits



Explanation:

* **Character.isDigit(ch)**: Checks if each character is a digit.
* If any character is not a digit, it returns false.

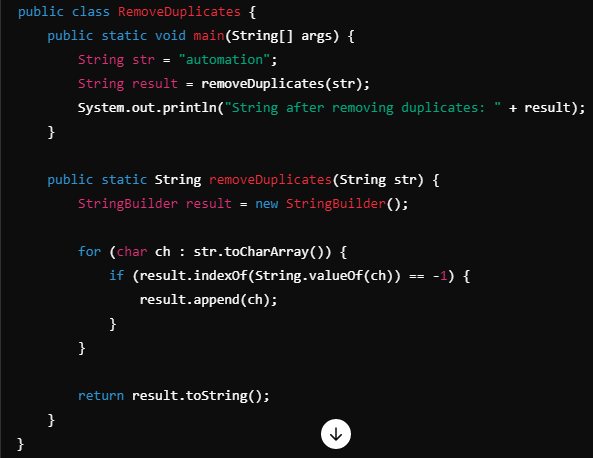
Count the occurrence of each character in a string



Explanation:

* **charCountMap.put(ch, charCountMap.getOrDefault(ch, 0) + 1)**: Increments the count of each character.
* **for (Map.Entry<Character, Integer> entry)**: Iterates through the map to print the count of each character.

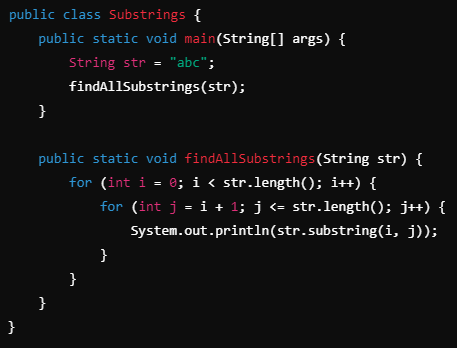
Remove duplicate characters from a string



Explanation:

* **result.indexOf(String.valueOf(ch)) == -1**: Checks if the character is already present in the result.
* If not present, the character is appended to the result.

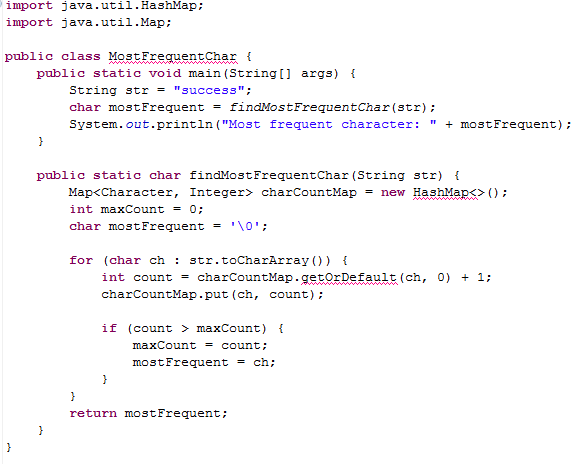
Find all substrings of a string



Explanation:

* **str.substring(i, j)**: Extracts all substrings starting from index i to j.
* Nested loops ensure that all possible substrings are printed.

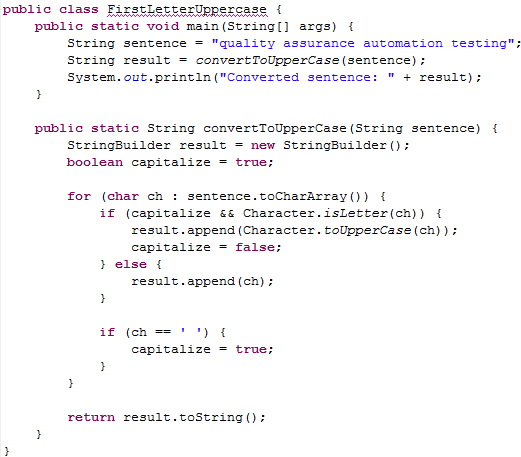
Find the most frequent character in a string



Explanation:

* **if (count > maxCount)**: Tracks the character with the highest frequency.
* Updates the most frequent character during iteration.

Convert the first letter of each word in a string to uppercase



Explanation:

* **boolean capitalize = true**: A flag to indicate when to capitalize a letter.
* **Character.toUpperCase(ch)**: Converts the character to uppercase if it is the first letter of a word.
* The flag is reset after every space character, allowing the first letter of the next word to be capitalized.