Задачи за упражнение - “Символни низове”

1. Напишете програма, която отпечатва всеки знак от символен низ на нов ред.
2. Напишете функция, която връща броя на буквите в символен низ (“aBc32a” -> 4).
3. Напишете функция, която намира първото срещане на знак в низ и го замества с друг знак (“bacd” ‘a’ ‘p’ -> “bpcd”)
4. Напишете функция, която проверява дали определен знак присъства в масив от знаци (“abcd” ‘b’ -> true).
5. Напишете функция `int length(char str[])`, която връща дължината на дадения символен низ, без да използва стандартните библиотечни функции.
6. Определете дали даден символен низ, въведен от конзолата, е палиндром (чете се както назад, така и напред) (“abcba”-> true, “abba” -> true, “asd” -> false).
7. Напишете програма, която извежда броя на гласните в символен низ, въведен от конзолата
8. Напишете програма, която премахва всички интервали от символен низ, въведен от конзолата.
9. Напишете програма, която при въведен от конзолата символен низ, го извежда обърнат (“abcd” -> “dcba”)
10. Преобразувайте всички главни букви в низ в малки букви и всички малки букви в главни букви.
11. Напишете функция `bool isSubstring(const char* str, const char* substr)`, която проверява дали `substr` е подниз на `str`.

1. Напишете програма, която намира и извежда първия неповтарящ се символ в символен низ (“abcabp” -> ‘c’)
2. Въведени са два символни низа, изведете ги конкатенирани, без да използвате стандартните библиотечни функции (“ab” “cd” -> “abcd”).
3. При даден символен низ и два низа (стар подниз и нов подниз), заменете всяко появяване на стария подниз с новия подниз (“Hi, I’m Vili, Hi, I’m Bogi” “Hi” “Bye” -> “Bye, I’m Vili, Bye, I’m Bogi”).
4. Проверете дали два дадени масива от знаци са анаграми един на друг (“heart” “earth” -> true)
5. Извършете “компресиране” на низове, използвайки броя на повтарящите се знаци. Например, масивът от знаци "aaabbccccd" трябва да стане "a3b2c4d1". А масивът “aaabbccaf” -> “a3b2c2a1f1”
6. Напишете функция, която премахва всички дублиращи се символи от символен низ. (“aabcdb” -> “abcd”)
7. При даден символен низ и число 'n', завъртете низа наляво с 'n' знака (“abcdef” 3 -> “defabc”)

1. Намерете лексикографски най-малката ротация на даден символен низ. Например за низа "bca" завъртанията са "bca", "cab" и "abc", а лексикографски най-малкият е "abc".
2. Създайте функция `void concatenate(char* destination, const char* source)`, която добавя изходния низ към целевия низ.
3. Даден е низ, съдържащ само знаците '(', ')', '{', '}', '[' и ']', напишете функция `bool isBalanced(const char s[])', която определя дали входът низът е валиден. Въведен низ е валиден, ако:

- Отворените скоби се затварят от същия тип скоби.
- Отворените скоби се затварят в правилния ред.
- Не съществуват несравними скоби.

Например "{[]}" и "()" са валидни, но "[)", "{[}]" и "{" не са.

4. Създайте функция `void longestPalindrome(char s[], char result[])`, която намира и връща най-дългия подниз, който е палиндром. Например, за входа "babad", функцията може да върне или "bab", или "aba".