

**Задачи за упражнение върху работа
със
СИМВОЛНИ НИЗОВЕ**

1. Напишете функция `isAlpha`, която по подаден символ казва дали символа е буква или не.
2. Напишете функция `isDigit`, която по подаден символ казва дали символа е цифра или не.
3. Напишете функциите `toUpper` / `toLowerCase`, който съответно преобразуват всички букви от малки в големи / големи в малки в един символен низ.
пример : `char* str = "abcd56ABCD";`
`toUpper(str) -> ABCD56ABCD` ; `toLowerCase(str) -> abcd56abcd`
4. Реализирайте функция `changeSpSymbol`, която по подадени символен низ (`s`), символ, който ще променяме (`c1`) и символ за замяна (`c2`) заменя всяко срещане на `c1` в `s` със `c2`
Пример :
`char* s = "a*b*c*d";`
`char c1 = '*';`
`char c2 = '1';`
`changeSpSymbol(s, c1, c2) -> "a1b1c1d"`
5. Имплементирайте функцията `atoi`, която по подаден валиден стринг съдържащ само цифри връща цялото число което се получава от този символите на този стринг.
*опитайте да разширите да работи и за стрингове с - отпред "-123"
Пример:
`const char* s = "123";`
`int res = atoi(s); cout << res; // 123`
6. Напишете функцията `isPalindrome`, която проверява дали даден стринг е палиндром. (прочетен отляво надясно и отдясно наляво е един и същ)
`const char *str = "a1b1a"; isPalindrome(str) -> true`

7. Имплементирайте функцията `reverse`, която обръща подаден стринг.

Пример :

```
char* str = "abcd";  
reverse(str);  
std::cout << str; // "dcba"
```

8. Реализирайте функция `dynamicConcat`, която конкатенира два низа, като обаче се грижи за паметта им. Може да използвате библиотечните `strlen`, `strcpy`, `strcat`.

9. Имплементирайте функция `compact`, която премахва всички гласни букви от един стринг, след премахването стрингът да има точно заделена памет спрямо броя на символите в него.

10. Реализирайте функцията `mostCommonAlpha`, която по подаден стринг отпечата на стандартния изход, коя е най-често срещаната буква в него и колко пъти се среща, като малки и големи букви считаме за еднакви.

Пример :

```
const char* str = "AaaaAaabBbCccc";  
mostCommonAlpha(str); // a 7
```

11. Напишете програма `dictionary`, която по предварително известен речник от думи и въведено изречение от потребителя казва колко от въведените думи се съдържат в речника.

Пример :

```
const char* dictionary[] = {"cats", "dogs"};  
Вход : it's raining cats and dogs  
Изход : 2
```

12. Реализирайте програмата "речта на Йода", която по въведено изречение в прав текст, отпечата на екрана примерен вариант за това как Йода би изрекал това изречение. (Йода / герой от *StarWars*, говори с разместен словоред).

Опитайте се да не заделяте повече памет от необходимата за всяка една от въведените думи от изречението. От програмата Ви се очаква при две различни `run` - вания с еднакво изречение за вход, да отпечата различен изход. (тоест ползвайте `rand` подходящо).

Вход : Young student the Force is strong with you!

Изход : the Young strong student is you! with Force

13. Напишете програмата crossword , която кара потребителя да въведе цяло положително число $N \leq 16$, след това на $N-1$ последователни реда да въведе думи с дължина **точно $N-1$ (защо?)** (разчитаме на верен вход)
След това програмата трябва да изведе броя на думите, които се срещат и в редовете, и в колоните.

Пример : 4

abc

bca

cba

Изход 1 (abc се среща в първия ред и в първата колона)

Пример : 4

abc

bcb

cbi

Изход 3

(abc се среща в първия ред и в първата колона,

bcb във втория ред и втората колона, cbi в третия ред с третата колона).