Выделите память для строки и заполните её любым способом. Создайте функцию, выполняющую задачу, указанную в варианте. Вызовите её из основной функции, получите её результат и выведите его. В случае, если результат – строка, исходная строка не должна измениться.

Вариант	Функция
1.	Написать функцию поиска LastIndexOf() последнего вхождения произвольной подстроки в строке.
2.	Написать функцию поиска IndexOf() первого вхождения произвольной подстроки в строке.
3.	Написать функцию, вводящую аналог операции “<” для сравнения двух строк (сравнение в алфавитном порядке).
4.	Написать функцию string_to_number(), проверяющую, является ли содержимое строки числом, и возвращающую это число в случае утвердительного ответа.
5.	Написать функцию, определяющую количество вхождений некоторой подстроки
6.	Написать функцию compare(), проверяющую совпадают ли две переданные ей строки
7.	Получить символ, который повторяется наибольшее количества раз. Если таких несколько, то вернуть массив из этих символов
8.	Написать функции, реализующие код Цезаря (и кодирование, и декодирование): буква заменяется на следующую за ней в алфавите (“test”->”uftu”)
9.	Написать функцию, проверяющую, является ли строка палиндромом
10.	Написать функцию, подсчитывающую число гласных
11.	Написать функцию endswith(), проверяющую, заканчивается ли строка переданным массивом символов
12.	Получить все цифры в строке в виде массива целых чисел
13.	Подсчитать количество знаков препинания (.,:;-) в исходной строке
14.	Написать функцию toupper(), преобразующую строку к верхнему регистру
15.	Подсчитать количество слов в строке. Словом будем считать последовательность, не содержащую цифр, имеющую длину не менее 3-х символов. Слова отделены друг от друга знаками препинания и пробелами.
16.	Подсчитать количество пар символов, в которых за согласной следует гласная.
17.	Написать функцию, подсчитывающую количество заглавных букв.
18.	Написать функцию concat(), объединяющую массив строк в одну строку
19.	Строку мысленно разбить на последовательности по 5 символов. Подсчитать количество пятерок символов, для которых количество гласных больше среднего для всей строки.
20.	Написать функцию trim(), удаляющую все пробелы в начале и конце строки
21.	Написать функцию reverse(), заменяющую порядок символов в строке на обратный
22.	Написать функцию, удаляющую лишние пробелы в тексте
23.	Написать функцию insert(), вставляющую подстроку в строку в место,

	обозначенное индексом index
24.	Написать функцию startswith(), проверяющую, начинается ли строка переданным массивом символов
25.	Определить число символов в строке, которые больше заданного символа ("больше" в смысле алфавитного порядка). Учесть, что регистр не важен.
26.	Подсчитать количество слов, которые начинаются и заканчиваются гласной.
27.	Определить, является ли строка допустимым идентификатором в C
28.	Получить индекс самого длинного слова в строке.
29.	В строке присутствуют скобки (). Проверить, что они корректно расставлены - для каждой открывающей есть закрывающая, и они расположены в правильном порядке. Пример неверной записи: ")()(", "(())())"
30.	Для всех слов в строке посчитать сумму значений кодов ASCII для символов, входящих в слово. Вернуть наибольшее значение.
31.	Проверить, идет ли в строке пробел после каждого знака препинания.
32.	Написать функцию tolower(), преобразующую строку к нижнему регистру
33.	Получить индекс последовательности с максимальным числом одинаковых символов. Если их несколько, вернуть индекс начала самой первой
34.	Подсчитать количество слов, в которых равное число гласных и согласных
35.	Написать функцию, которая принимает строку и возвращает новую, в которой все гласные переставлены в начало массива
36.	Получить индекс максимального по длине участка идущих подряд согласных

Пример кода:

```
#include <stdio.h>
 #include <stdlib.h>

 int get_len(const char* str)
 {
 //на вход функция получает строку и вычисляет её длину
 int len = 0;
 while(str[len]){ // аналогично str[len]!='\0' - проверка на символ конца строки
 len++;
 }
 return len;
 }
```

```
char* space2zero(const char* str)
{
//функция заменяет пробелы на 0
int len= get_len(str);
char* s = (char*)malloc(sizeof(char)*(len+1));
int i;
for (i = 0;i<=len;i++){
if(str[i]==' '){
s[i]='0'; // !заменяем на символ '0', а не на число 0
}else{
s[i]=str[i];
}
}

return s;
}


int main()
{
// Определяем длину строки-литерала
const char* str = "Hello World";
printf("%s\n", str);
int len = get_len(str);
printf("%d\n", len);
char* zero_str=space2zero(str)
printf("%s\n", zero_str);

// Определяем длину строки, введенную пользователем
char buff[256];
scanf("%s", &buff);
len = get_len(buff);
printf("%d\n", len);
free(zero_str);
return 0;
}
```