

Индивидуальное задание №4

Строки, указатели, массивы символов

В задаче А головная программа должна демонстрировать возможности работы с разработанной Вами функцией для работы со строками. **Запрещено использовать стандартные функции по обработке строк.**

В задаче В предполагается, что длина входной строки не превосходит 300 символов, она вводится с консоли. Вам необходимо вывести на консоль либо преобразованную строку, либо требуемые в условии данные.

Использовать класс string нельзя!

Вариант 1

А. Написать собственную реализацию стандартной функции `strcat`.

Функция `strcat` описана в заголовочном файле `<string.h>` как

```
char *_strcat (char *strDestination, const char *strSource);
```

и предназначена для добавления символов из `strSource` к концу строки `strDestination` (сцепление строк `strDestination` и `strSource`). Никаких проверок на переполнение выделенной памяти не производится. Функция возвращает адрес `strDestination`.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти слово, в котором число различных символов минимально. Если таких слов несколько, найти первое из них.

Вариант 2

А. Написать собственную реализацию стандартной функции `strcmp`.

Функция `strcmp` описана в заголовочном файле `<string.h>` как

```
int _strcmp (char *str1, const char *str2);
```

и предназначена для сравнения строк `str1` и `str2`.

Возвращаемое значение	Отношение между строками
-1	<code>str1</code> лексикографически меньше, чем <code>str2</code>
0	<code>str1</code> эквивалентна <code>str2</code>
1	<code>str1</code> лексикографически больше, чем <code>str2</code>

В. Строка состоит из слов, разделенных ровно одним пробелом, пробелов перед первым и после последнего слова нет. Поместить в начало строки слова, содержащие только цифры, а затем – все остальные слова. Порядок слов внутри заданных групп не должен изменяться.

Вариант 3

А. Написать собственную реализацию стандартной функции `strstr`.

Функция `strstr` описана в заголовочном файле `<string.h>` как

```
char *_strstr (const char *string, const char *strCharSet);
```

и предназначена для поиска строки strCharSet в строке string. Возвращается указатель на начальный символ первого вхождения strCharSet в строку string или NULL, если string не содержит strCharSet.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти количество слов, содержащих только символы латинского алфавита, а среди них – количество слов с равным числом гласных и согласных букв.

Вариант 4

А. Написать собственную реализацию стандартной функции strcpy.

Функция strcpy описана в заголовочном файле <string.h> как

```
char *_strcpy (char *strDestination, const char *strSource);
```

и предназначена для копирования строки strSource в строку strDestination. Никаких проверок на переполнение выделенной памяти не производится. Функция возвращает адрес Destination.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти слово, символы в котором идут в строгом возрастании их кодов. Если таких слов несколько, найти первое из них.

Вариант 5

А. Написать собственную реализацию стандартной функции itoa.

Функция itoa описана в заголовочном файле <stdlib.h> как

```
char *_itoa (int value, char *string, int radix);
```

и предназначена для преобразования целого числа value в его символьное представление string в системе счисления с основанием radix. Значение radix лежит в интервале от 2 до 36, старшая цифра в 36-ричной системе счисления – 'Z'. Функция возвращает адрес string.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти слово, в котором число различных символов максимально. Если таких слов несколько, найти последнее из них.

Вариант 6

А. Написать собственную реализацию стандартной функции atoi.

Функция atoi описана в заголовочном файле <stdlib.h> как

```
int _atoi (const char *string);
```

и предназначена для преобразования строки string, содержащей символьное представление целого числа в десятичной системе счисления, к этому числу.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти слово, один и тот же символ в котором встречается максимальное число раз (в рамках всей строки). Если таких слов

несколько, найти первое из них.

Вариант 7

А. Написать собственную реализацию стандартной функции `strchr`.

Функция `strchr` описана в заголовочном файле `<string.h>` как

```
char *_strchr (const char *string, int c);
```

и предназначена для поиска в нуль-терминированной строке `string` символа `c` (признак конца строки также участвует в поиске). Возвращается указатель на начальный символ первого вхождения `c` в строку `string` или `NULL`, если `string` не содержит этого символа.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти количество слов, представляющих собой правильные записи дат в формате `"dd/mm/yyyy"`.

Вариант 8

А. Написать собственную реализацию стандартной функции `strrchr`.

Функция `strrchr` описана в заголовочном файле `<string.h>` как

```
char *_strrchr (const char *string, int c);
```

и предназначена для поиска в нуль-терминированной строке `string` символа `c` (признак конца строки также участвует в поиске). Возвращается указатель на начальный символ последнего вхождения `c` в строку `string` или `NULL`, если `string` не содержит этого символа.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Найти слово, состоящее только из различных символов. Если таких слов несколько, найти первое из них.

Вариант 9

А. Написать собственную реализацию стандартной функции `strncat`.

Функция `strncat` описана в заголовочном файле `<string.h>` как

```
char *_strncat (char *strDest, const char *strSource, size_t count);
```

и предназначена для добавления не более чем `count` символов из `strSource` к концу строки `strDest`. Никаких проверок на переполнение выделенной памяти не производится. Функция возвращает адрес `strDest`.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Среди слов, состоящих только из цифр, найти слово-палиндром. Если таких слов больше одного, найти второе из них.

Вариант 10

А. Написать собственную реализацию стандартной функции `strspn`.

Функция `strspn` описана в заголовочном файле `<string.h>` как

```
int _strspn (const char *string, const char *strCharSet);
```

и возвращает длину начальной подстроки из `string`, содержащей только символы из множества `strCharSet`.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Среди слов, состоящих только из цифр, найти слово, содержащее максимальное число нулей. Если таких слов больше одного, найти предпоследнее из них.

Вариант 11

А. Написать собственную реализацию стандартной функции `strcspn`.

Функция `strcspn` описана в заголовочном файле `<string.h>` как

```
int _strcspn (const char *string, const char *strCharSet);
```

и возвращает длину начальной подстроки из `string`, не содержащей символов из множества `strCharSet`.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Если первый символ слова является строчной латинской буквой, заменить этот символ на соответствующую прописную букву.

Вариант 12

А. Написать собственную реализацию стандартной функции `strpbrk`.

Функция `strpbrk` описана в заголовочном файле `<string.h>` как

```
char *_strpbrk(char *string, const char *strCharSet );
```

сканирует строку на наличие символов из множества `strCharSet` и возвращает указатель на первое вхождение любого символа из `strCharSet` в строку `string` или `NULL`, если две строки параметров не содержат общих символов.

В. Строка состоит из слов, разделенных одним или несколькими пробелами. Среди слов, состоящих только из цифр, найти слово, содержащее максимальное число единиц. Если таких слов больше одного, найти предпоследнее из них.