3 СИМВОЛЬНАЯ ОБРАБОТКА

Исходный текст должен вводиться пользователем с экрана в виде одной или нескольких строк. На экране необходимо *сохранить протокол работы*: исходный и преобразованный текст. После завершения обработки текста, и вывода результата, необходимо *предоставить пользователю возможность продолжения работы* с другим текстом. При выполнении работы алгоритмические подзадачи *выделять в отдельные функции*. Алгоритмы должны работать *независимо от регистра символов*.

Для ввода текста с пробелами использовать функцию gets(char *str). При выполнении работы допускается использовать следующие функции, работы со строками, объявленные в заголовочном файле string.h:

`char *strcat(char *dest, const char *src);`	дописывает строку src в конец строки dest
`char *strncat(char *dest, const char *src, size_t n);`	дописывает не более n начальных символов строки src (или всю src, если её длина меньше) в конец строки dest
`char *strchr(const char *s, int c);`	возвращает адрес символа c в строке s, начиная с головы, или NULL, если строка s не содержит символ c
`char *strrchr(const char *s, int c);`	возвращает адрес символа c в строке s, начиная с хвоста, или NULL, если строка s не содержит символ c
`int strcmp(const char *str1, const char *str2);`	лексикографическое сравнение строк (возвращает "0", если строки одинаковые, положительное, если первая строка больше, и отрицательное, если меньше)
`int strncmp(const char *str1, const char *str2, size_t n);`	лексикографическое сравнение первых n байтов строк
`char *strcpy(char *toHere, const char *fromHere);`	копирует строку из одного места в другое
`char *strncpy(char *toHere, const char *fromHere, size_t n);`	копирует до n байт строки из одного места в другое
`size_t strlen(const char *);`	возвращает длину строки

Так же полезными в выполнении задания могут оказаться следующие функции работы с символами, объявленные в заголовочном файле ctype.h:

`int islower(int c);`	возвращает true (значение отличное от 0), если c - символ нижнего регистра, 0 в противном случае
`int isupper (int c);`	возвращает true (значение отличное от 0), если c - символ верхнего регистра, 0 в противном случае
`int tolower (int c);`	если c - символ верхнего регистра, возвращает его версию в нижнем регистре; иначе просто возвращает исходный символ c
`int toupper (int c);`	если c - символ нижнего регистра, возвращает его версию в верхнем регистре; иначе просто возвращает исходный символ c

Варианты

3.1. Записать звучание английского (немецкого, французского) текста русскими буквами (использовать правила транскрипции).

3.2. Вставить во все слова текста символ переноса в позициях, где по правилам русского языка допустим перенос.

3.3. Записать заданное натуральное число N русскими словами (семнадцать, триста семьдесят один, тысяча пятьсот сорок четыре и т. д.).

3.4. Организовать принцип «Поиск и замена слов», т.е. найти в тексте все слова заданного вида, и заменить эти слова на заданное слово. Выделить в исходном тексте слова, равные заданному, а в результирующем тексте «новые» слова.

3.5. Организовать принцип «Поиск и замена фрагмента», т.е. найти в тексте все слова, которые содержат фрагмент заданного вида, и заменить этот фрагмент на новый. Выделить в исходном и результирующем текстах измененные слова и фрагменты.

3.6. Дан текст. Преобразовать его следующим образом: удалить все повторы слова кроме первого вхождения, для всех слов, которые встречаются в этом тексте более одного раза. Вывести преобразованный текст, оставшиеся первые вхождения удалённых слов выделить скобками.

3.7. Найти в тексте слова, которые содержат хотя бы одну заданную букву. Если слово начинается на эту букву, то оставить его без изменения, если заканчивается, то слово «перевернуть», если «внутри», то заменить заданную букву на новую заданную букву. Все измененные слова выделить в тексте скобками.

3.8. Найти в тексте все глаголы, имеющие неопределенную форму. Выделить их в тексте скобками.

3.9. Найти в тексте все слова, которые содержат хотя бы один фрагмент заданного вида. Подсчитать их количество и выделить в исходном тексте скобками как слова, так содержащиеся в них фрагменты.

3.10. Подсчитать среднюю длину слова в заданном тексте. Найти в тексте все слова, длина которых больше средней длины и выделить их скобками в исходном тексте.

3.11. Получить строку символов, являющуюся записью числа в десятичной системе счисления, из строки символов, являющейся записью этого числа в шестнадцатеричной системе счисления. Проверять корректность задания исходных данных, для преобразования в строковое десятичное значение не использовать стандартные функции.

3.12. Дан текст. Составить новый текст, в котором слова упорядочены по длине (слово минимальной длины – первое по порядку, максимальной длины – последнее по порядку), среди слов одинаковой длины сохранить порядок следования, как в исходном тексте.

3.13. Получить строку символов, являющуюся записью числа в десятичной системе счисления, из строки символов, являющейся записью этого числа в системе счисления с основанием от 2 до 9. Проверять корректность задания исходных данных, для преобразования в строковое десятичное значение не использовать стандартные функции.

3.14. Найти в тексте все слова, начинающиеся на заданную букву русского алфавита. Найденные слова «перевернуть». Определить их количество и выделить в исходном тексте скобками.

3.15. Найти в тексте слова, которые содержат хотя бы один слог «НА», «ПРИ», «ЗА», выделить слова и искомые слоги в тексте скобками.

3.16. Сложить два числа, записанных римскими цифрами (не переводя в десятичную систему). Результат получить в виде числа, записанного римскими цифрами. Проверить правильность записи римских чисел.

3.17. Перевести число, записанное римскими цифрами, в десятичную систему счисления, и наоборот. Проверить правильность записи римского числа.

3.18. Дан текст. Получить другой текст, составленный из слов исходного текста, таким образом, что в каждом четном слове буквы упорядочиваются строго по алфавиту, а в каждом нечетном буквы упорядочиваются в порядке обратном алфавитному.

3.19. Организовать принцип «Форматирование абзаца» для заданного текста по выбору пользователя (по ширине окна, по центру, по левому и правому краю). Форматирование строки (абзаца) заключается в том, что между ее отдельными словами дополнительно вносятся пробелы. Пример форматирования по ширине:

необходимо равномерно вставить пробелы между словами таким образом, чтобы первое слово было прижато к левому краю окна, а последнее слово строки сдвинулось к ее правому краю.

3.20. Провести частотный анализ текста. Построить столбиковую гистограмму, отражающую количество слов в тексте, начинающихся на буквы заданного алфавита. Буквы выводить в алфавитном порядке. Заглавные и строчные буквы считать одним символом.

3.21. Провести частотный анализ текста. Построить столбиковую гистограмму, отражающую частоту появления символов в заданном тексте. Символы выводить в порядке убывания частоты. Заглавные и строчные буквы считать одним символом.

3.22. Найти в тексте все слова, удовлетворяющие заданной пользователем маске (маска содержит буквы и символ *, который заменяет любую букву). Подсчитать их количество и выделить в исходном тексте скобками.

3.23. Ввести два произвольных текста. Проверить, содержат ли они одинаковую последовательность слов (между словами может быть различное число знаков препинания).

3.24. Найти в тексте все фрагменты, удовлетворяющие заданной пользователем маске (маска содержит буквы и символ *, который заменяет любую букву). Подсчитать количество найденных фрагментов и выделить их в тексте скобками.

3.25. Найти в тексте все слова, которые являются палиндромами, т.е. читаемыми в прямом и обратном направлении (например, слова «шалаш», «кок» - палиндромы). Выделить их в тексте скобками.

3.26. Дан текст, записанный азбукой Морзе, буквы разделены пробелом, слова - двумя или более пробелами. Перевести его в обычную запись. Проверять корректность задания исходных данных.

3.27. Найти в тексте слова, начинающиеся и заканчивающиеся на заданные буквы русского алфавита. Определить их количество и выделить в исходном тексте скобками.

3.28. Определить, является ли заданный текст корректным синтаксисом определения целочисленной переменной в языке СИ.

3.29. Дан текст. Проверить, упорядочен ли он по длине слов (слово минимальной длины – первое по порядку, максимальной длины – последнее по порядку).

3.30. Дан текст. Получить строку-алфавит, т.е. последовательность неповторяющихся символов, из которых построен исходный текст. Результирующая строка-алфавит должна быть упорядочена по алфавиту.