Задачи к лабораторной работе на тему
«Программирование на языке С с использованием нуль-терминальных строк»

Цель работы: изучить особенности написания программ на языке *С* с использованием нуль-терминальных строк.

Указания к выполнению работы

При решении всех задач программы должны быть полностью совместимы с языком программирования С, а именно:

- вводимые пользователем данные необходимо представлять в виде нуль-терминальных строк;
- недопустимо использовать строковые функции языка С++;
- недопустимо использовать распределение динамической памяти языка С++;
- недопустимо использовать стандартный потоковый ввод-вывод `iostream`.

При решении задачи 1 нужно использовать как минимум три различных функции `string.h`. Недопустимо использовать операции индексации и обращение к отдельным символам массива через указатели.

При определении функции задачи 3 недопустимо использовать строковые функции и операцию индексации. Обеспечить возможность ввода пользователем данных, вызов описанной функции и вывод результата ее работы.

Индивидуальные варианты заданий

Задача 1

1. Даны две строки `str1` и `str2`. Если строки равны лексикографически, то вывести первую строку, все символы которой привести

к верхнему регистру, в противном случае, вывести вторую строку, все символы которой привести к нижнему регистру.

2. Даны две строки `str1` и `str2`. Если строки не равны лексикографически, то вывести большую строку, при этом все символы строки привести к нижнему регистру, в противном случае, вывести строку, все символы которой привести к верхнему регистру.

3. Даны две строки `str1` и `str2`. Если строки не равны по длине, то вывести строку с меньшей длиной. Если строки совпадают лексикографически, то вывести строку в верхнем регистре.

4. Даны две строки `str1` и `str2`. Если строки совпадают по длине, но не равны лексикографически, то вывести меньшую строку, при этом все символы строки привести к верхнему регистру, в противном случае, вывести новую строку, состоящую из всех символов строки `str2`, следующих за всеми символами строки `str1`.

5. Даны две строки `str1` и `str2`. Получить и вывести на экран новую строку, состоящую из частей: 1) все символы строки `str1`, записанные в верхнем регистре, 2) все символы строки `str2` в обратном порядке.

6. Даны две строки `str1` и `str2`. Получить и вывести на экран новую строку, состоящую из частей: 1) все символы первой строки, записанные в верхнем регистре, 2) знак пробела, 3) все символы второй строки, записанные в нижнем регистре.

7. Даны две строки `str1` и `str2`. Получить и вывести на экран новую строку, состоящую из частей: 1) все символы большей (лексикографически) строки, 2) точка, 3) все символы меньшей строки, записанные в верхнем регистре.

8. Даны две строки `str1` и `str2`. Если строки равны по длине, то вывести новую строку, состоящую из частей: 1) последовательности символов первой строки, 2) последовательности символов второй строки. Если строки совпадают лексикографически, то вывести строку в верхнем регистре.

9. Даны две строки `str1` и `str2`. Если строки имеют одинаковую длину, но не совпадают лексикографически, то вывести большую строку, в противном случае изменить порядок следования символов строки `str1` на обратный.

10. Даны две строки `str1` и `str2`. Получить новую строку, состоящую из частей: 1) все символы второй строки, записанные в обратном порядке, 2) знак запятой, 3) все символы первой строки, записанные в нижнем регистре.

11. Даны две строки `str1` и `str2`. Если строки не равны, то в строку `str1` скопировать строку `str2`, в противном случае получить строку, состоящую из частей: 1) знак вопроса, 2) все символы первой строки, 3) восклицательный знак.

12. Даны две строки `str1` и `str2`. Получить и вывести на экран новую строку, в которой все символы записаны в нижнем регистре и состоящую из: символов строки `str2`, начинающихся с последовательности символов, совпадающих со строкой `str1` до конца строки.

13. Даны две строки `str1` и `str2`. Получить и вывести на экран новую строку, в которой все символы записаны в обратном порядке и состоящую из символов первой строки, начиная с последовательности символов, совпадающих с `str2` до конца строки.

14. Даны две строки `str1`, `str2` и символ `symbol`. Получить и вывести на экран новую строку, состоящую из частей: 1) подстроки первой строки, начинающейся с первого вхождения символа `symbol` в `str1` до ее конца 2) второй строки, символы которой записаны в обратном порядке.

15. Даны три строки `str1`, `str2` и `str3`. Получить и вывести на экран новую строку, состоящую из частей: 1) все символы первой строки, начиная с последовательности символов, совпадающих с `str3`, 2) знак пробела, 3) все символы второй строки, записанные в обратном порядке.

16. Даны две строки str1, str2 и целое n. Если строки имеют одинаковую длину, но не совпадают лексикографически, то вывести меньшую строку. Если строки совпадают полностью, то получить и вывести на экран новую строку, состоящую из частей: 1) n первых символов строки, 2) знак двоеточия, 3) перевернутая копия строки.

17. Даны две строки str1, str2 и целое n. Получить и вывести на экран новую строку, состоящую из частей: 1) n первых символов первой строки, записанные в нижнем регистре, 2) знак тире, 3) все символы второй строки, записанные в верхнем регистре.

18. Даны две строки str1, str2 и целое n. Если длины строк не равны, то в строку str1 скопировать n первых символов строки str2, в противном случае получить строку, состоящую из частей: 1) знак вопроса, 2) n первых символов строки str1, 3) вопросительный знак.

19. Даны две строки str1, str2 и целое n. Если первые n символов строк не совпадают, то заменить строку str1 строкой str2, в противном случае, добавить в конец строки str2 ее перевернутую копию.

20. Даны две строки str1, str2 и целое n. Если первые n символов строк равны, то вывести подстроку строки str1, начиная с n-го символа до конца строки.

21. Даны две строки str1 и str2. Получить новую строку, в которой записаны все символы строки str1 до последнего вхождения строки str2 в str1.

22. Даны две строки str1, str2 и символ symbol. Заменить в строке str1 все символы, начинающиеся с первого символа symbol до конца строки на все символы строки str2, начинающиеся с последнего символа symbol до конца строки.

23. Даны три строки str1, str2, str3 и символ symbol. Заменить в строке str1 все символы, начинающиеся с подстроки str3 до конца строки

на все символы строки str2, начинающиеся с символа symbol до конца строки.

24. Даны три строки str1, str2 и str3. Заменить в строке str2 все символы, начинающиеся с подстроки str1 до конца строки на строку str3, символы которой записаны в обратном порядке.

25. Даны три строки str1, str2, str3 и символ symbol. Заменить в строке str1 все символы, начинающиеся с подстроки str3 до конца строки на подстроку строки str2, начинающуюся с символа symbol до конца строки.

26. Даны две строки str1 и str2, символ symbol и целое n. Заменить в строке str1 n символов, начиная с последнего символа symbol на подстроку строки str2, начинающуюся с символа symbol и заканчивающуюся концом строки str2.

27. Даны две строки str1 и str2, символ symbol и целое n. Составить и вывести на экран новую строку, состоящую из частей: 1) все символы строки str1, начиная с последнего символа, совпадающего с symbol, заканчивая концом строки str1, 2) n первых символов строки str2, начиная с первого символа, совпадающего с symbol.

28. Даны три строки str1, str2, str3, символ symbol и целое n. Если первые n символов строки str1 не совпадают с символами строки str2 (без учета регистра), то заменить строку str1 на подстроку строки str3, начинающуюся с последнего в ней символа, совпадающего с symbol, до конца строки (с учетом регистра).

29. Дана строка str1 и символ symbol. Если в строке str1 символ symbol встречается более одного раза, вывести подстроку строки str1, от начала строки до последнего вхождения в нее символа symbol.

30. Даны две строки str1 и str2 и целое n. Если первые n символов строк str1 и str2 совпадают (без учета регистра), то в строке str1 записать в обратном порядке все символы, начиная с (n + 1)-го.

31. Дана строка `str1` и символ `symbol`. Если в строке `str1` символ `symbol` встречается только один раз, выведите его индекс. Если `symbol` встречается два и более раз, выведите индекс его первого и последнего появления. Если `symbol` в данной строке не встречается, ничего не выводите.

32. Дана строка `str1` и символ `symbol`, который встречается в `str1` как минимум два раза. Разверните последовательность символов, заключенную между первым и последним появлением `symbol`, в противоположном порядке.

33. Дана строка `str1` – строка, длиной не более 1000 символов, состоящая из цифр (без ведущих нулей). Чтобы предсказать судьбу человека, нумеролог берет время жизни человека в секундах, затем складывает все цифры этого числа. Если полученное число состоит более чем из одной цифры, операция повторяется, пока в числе не останется одна цифра. Затем по полученной цифре и числу операций, необходимых для преобразования числа в цифру нумеролог предсказывает судьбу человека. Выведите два числа через пробел: полученную цифру из числа, представленного строкой `str1` и количество преобразований.

34. Дана строка `str1` – строка, длиной не более 10^5 символов, содержащую только большие и маленькие буквы английского алфавита. Руны — это древние магические знаки, которые наши предки использовали как буквы. Говорят, что рунные знаки обладают магическими свойствами, а при сложении рун в слова их магическая сила многократно возрастает. Если кузнец изготовит доспехи и начертит там определенные руны в определенном порядке, то доспехи будут наделены необычайными магическими силами. Для того, чтобы стать обладателем таких доспехов достаточно просто принести кузнецу начертания этих рунных знаков. А вот, чтобы стать обладателем рунного знака приходилось немало потрудиться. Воины добывали начертания рун других языков и наречий в боях или получали их в качестве наград в благодарность за оказанные услуги. Но так или иначе и в этом деле развелись жулики. По подозрениям ученых кузнец Игнатус Мошеникус изготавливал

благородным воинам фальшивые рунные слова. Из древних преданий ученым стало достоверно известно, что каждая руна записывается из двух, трех или четырех английских букв. Причем первая буква рунного слова всегда записывается как заглавная, а все остальные являются маленькими. Ученые перевели несколько, выкованных этим кузнецом, рунных слов на английский язык и теперь нуждаются в Вашей помощи. Проверьте, является ли приведенное слово рунным.

35. Дана строка str1 – строка, длиной не более 1000 символов, содержащую только маленькие буквы английского алфавита и целое число k ≠ 0, |k| < 100001. Пусть задана строка $s = s_1 s_2 \ldots s_n$. Назовем ее k-ой (k > 0) степенью s^k строку $s^k = s_1 s_2 \ldots s_n s_1 s_2 \ldots s_n \ldots\ldots s_1 s_2 \ldots s_n$ (k раз). Например, третьей степенью строки abc является строка abcabcabc. Корнем k степени из строки s называется такая строка t (если она существует), что $t^k = s$. Напишите программу, находящую степень строки или корень из нее. Если k > 0, то необходимо найти k-ую степень строки s, если k < 0, то необходимо найти корень степени |k| из s. Если длина ответа превосходит 1023 символа, выведите только первые 1023 символа. Если искомой строки не существует — выведите NO SOLUTION.

Задача 2

1. Дан текст, в котором встречаются буквы "и" и "т". Определить какая из них встречается позже (при просмотре текста слева направо). Если таких букв несколько, то должны учитываться последние из них.

2. Дан текст, в котором имеется одна буква "а" и одна буква "о". Определить, сколько символов находится между буквами "а" и "о".

3. Дан текст, в котором имеется одна буква "к" и как минимум одна буква "о". Вывести текст, расположенный между буквой "к" и крайней справа буквой "о".

4. Дан текст, предложения в котором разделены точкой. Определить сколько в тексте предложений.

5. Дан текст, слова в котором разделены одинарным символом пробела (символ "-" в тексте отсутствует). Верно ли, что число слов в предложении больше трех?

6. Дан текст. Определить верно ли, что букв "а" в тексте больше, чем букв "о"?

7. Дан текст, слова в котором разделены одинарным символом пробела (символ "-" в тексте отсутствует). Определить сколько в тексте слов.

8. Дан текст. Определить сколько раз в тексте встречается его последняя буква.

9. Дан текст, предложения в котором разделены точкой. Определить, сколько раз в первом предложении текста встречается буква "а" (предусмотреть вариант, что в первом предложении такой буквы может не быть).

10. Дан текст. Напечатать все символы текста, расположенные между первой и второй запятой. Если второй запятой нет, то надо вывести все символы, расположенные после первой запятой. Если запятых нет - вывести весь текст.

11. Дан текст и символ `symbol`. Удвоить каждое вхождение символа `symbol` в тексте.

12. Дан текст и какая-то буква. Определить количество заданных букв в тексте, предшествующих первой запятой (если запятая отсутствует - определить общее количество заданных букв).

13. Дан текст. Определить количество цифр, встречающихся в тексте.

14. Дан текст. Определить верно ли, что в нем есть четыре подряд расположенных одинаковых символа.

15. Дан текст. Определить сколько в тексте одинаковых соседних букв.

16. Дан текст. Определить первую и последнюю пару одинаковых соседних символов в тексте.

17. Даны два слова, разделенные пробелом. Определить, сколько начальных букв первого слова совпадает с начальными буквами второго слова.

18. Дан текст, слова в котором разделены одинарным символом пробела. Вывести слово, содержащее удвоенную букву "н" (если подобных слов несколько - вывести последнее из них).

19. Дан текст, слова в котором разделены одинарным символом пробела (символ "-" в тексте отсутствует). Вывести все слова, в которых встречается буквосочетание "ее".

20. Дан текст. Проверить, правильно ли записаны буквосочетания "ча" и "ща" и исправить ошибки.

21. Дан текст. Определить частоту появления букв "а" в нем. Частота вычисляется как отношение количества данных символов в тексте к длине всего текста (пробелы учитываются, а символ конца строки не учитывается).

22. Дан текст. Определить, сколько в нем гласных букв.

23. Дан текст, слова в котором разделены произвольным количеством пробелов. Удалить повторяющиеся пробелы между отдельными словами (оставляя по одному пробелу).

24. Дан текст, слова в котором разделены одинарным символом пробела, и какая-то буква. Посчитать количество слов, начинающихся на заданную букву.

25. Дан текст, слова в котором разделены одинарным символом пробела. Вывести все слова текста, которые начинаются и заканчиваются на одну и ту же букву.

26. Дан текст, слова в котором разделены одинарным символом пробела, и какая-то буква. Вывести все слова, в которых встречается эта буква.

27. Дан текст, слова в котором разделены одинарным символом пробела (в тексте могут встречаться запятые), и какая-то буква. Вывести все слова текста, у которых последняя буква не совпадает с заданной.

28. Дан текст, слова в котором разделены одинарным символом пробела (знаки пунктуации отсутствуют). Определить сколько в тексте слов-перевертышей.

29. Дан текст, слова в котором разделены одинарным символом пробела (знаки пунктуации отсутствуют). Вывести все симметричные слова текста, отличные от первого.

30. Дан текст, слова в котором разделены одинарным символом пробела (знаки пунктуации отсутствуют), какая-то буква и целое n. Вывести все слова текста, в которых заданная буква встречается ровно n раз.

31. Дан текст на русском языке, слова в котором разделены одинарным символом пробела (знаки пунктуации отсутствуют). Написать программу "Телеграф", которая выводит на экран заданный текст в виде последовательности точек и тире (Азбука Морзе).

32. Дан текст, слова в котором разделены одинарным символом пробела (знаки пунктуации отсутствуют). На планете Роботов очень не любят десятичную систему счисления, поэтому они попросили Вас написать программу, которая заменяет все встречающиеся в тексте числа на эти же числа, но в двоичной системе счисления. Гарантируется, что во всех числах нет ведущих нулей.

33. Капитан Флинт зарыл клад на Острове сокровищ. Он оставил описание, как найти клад. Описание состоит из строк вида: "North 5", где слово – одно из "North", "South", "East", "West", – задает направление движения, а число – количество шагов, которое необходимо пройти в этом направлении. Напишите программу, которая по описанию пути к кладу определяет точные координаты клада, считая, что начало координат находится в начале пути, ось OX направлена на восток, ось OY – на север.

На вход подается последовательность строк указанного формата. Гарантируется, что числа не превосходят 10^8. Необходимо вывести координаты клада – два целых числа через пробел. Гарантируется, что эти числа не превосходят 10^8.

34. Для того чтобы выходить в Интернет, каждому компьютеру присваивается так называемый IP-адрес. Он состоит из четырех целых чисел в диапазоне от 0 до 255, разделенных точками. В следующих трех строках показаны три правильных IP-адреса:

127.0.0.0

192.168.0.01

255.00.255.255

Напишите программу, которая определяет, является ли заданная строка правильным IP-адресом.

Входные данные - строка длиной не более 15 символов, которая включает цифры и ровно три точки.

Если строка является правильным IP-адресом, необходимо вывести 1, иначе 0.

35. Мескленитs отправились в экспедицию. Однажды руководителю экспедиции потребовалось отправить на разведку специальный отряд, состоящих из лучших мескленитов. Для этого он выстроил всю команду в шеренгу. Цвет панциря каждого мескленита обозначается заглавной латинской буквой (от "A" до "Z"). В целях экономии времени руководитель собирается выбрать из шеренги несколько подряд стоящих. Кроме того, он считает, что разведка будет более удачной, если выбранный отряд будет симметричен по цветам панцирей. Например, отряд "RGBGR" будет симметричным, а отряд "RGRB" – нет. Требуется выбрать из шеренги мескленитов максимально возможный отряд, удовлетворяющий данным условиям.

Входные данные - строка, длина которой не превосходит 255 символов – цвета мескленитов в шеренге.

Выходные данные – строка - выбранный отряд мескленитов. Если возможных вариантов ответа несколько, то требуется вывести находящийся ближе к началу шеренги.

Задача 3

1. Написать функцию, возвращающую длину строки. Функция должна иметь прототип:

```
int strlength(const char* str),
```

где `str` – строка-источник.

2. Написать функцию, выполняющую конкатенацию строк и возвращающую результат объединения. Функция должна иметь прототип:

```
char* strconcat(const char* str1, const char* str2),
```

где `str1` – строка-источник, `str2` – присоединяемая строка.

3. Написать функцию, возвращающую индекс заданного символа в строке, поиск ведется от начала строки. Функция должна иметь прототип:

```
int charpos(const char* str, const char c),
```

где `str` – строка-источник, `c` – искомый символ.

4. Написать функцию, возвращающую индекс заданного символа в строке, начиная с заданной позиции. Функция должна иметь прототип:

```
int charpos(const char* str, const char c, unsigned i),
```

где `str` – строка-источник, `c` – искомый символ, `i` – индекс начала поиска в строке. Если `i` не задан – производить поиск с 1 символа.

5. Написать функцию, возвращающую подстроку заданной строки, расположенную между заданными индексами. Функция должна иметь прототип:

```
char* substri(const char* str, unsigned i, unsigned j),
```

где `str` – строка-источник, `i` – начальный индекс подстроки, `j` – конечный индекс подстроки.

6. Написать функцию, возвращающую подстроку, начинающуюся с символа поиска до конца строки. Функция должна иметь прототип:

```
char* strcpos(const char* str, const char c),
```

где `str` – строка-источник, `c` – искомый символ.

7. Написать функцию, возвращающую строку-источник с заменой каждого вхождения заданного символа другим заданным символом. Функция должна иметь прототип:

```
char* strcreplc(const char* str, const char c, const char r),
```

где `str` – строка-источник, `c` – символ, который надо заметить символом `r`.

8. Написать функцию, возвращающую строку, содержащую символы исходной строки, расположенные в обратном порядке. Функция должна иметь прототип:

```
char* strreverse(const char* str),
```

где `str` – строка-источник.

9. Написать функцию, возвращающую индекс начала подстроки в строке, поиск в строке ведется с первой позиции. Функция должна иметь прототип:

```
int strpos(const char* str, const char* substr),
```

где `str` – строка-источник, `substr` – искомая подстрока.

10. Написать функцию, возвращающую индекс начала подстроки в строке, поиск в строке ведется с заданной позиции. Функция должна иметь прототип:

```
int strpos(const char* str, const char* substr, unsigned i),
```

где `str` – строка-источник, `substr` – искомая подстрока, `i` – индекс начала поиска.

11. Написать функцию для изменения содержимого строки по следующему принципу: 1) разрежьте строку на две равные части если длина строки — четная, а если длина строки нечетная, то длина первой части должна быть на один символ больше; 2) переставьте эти две части местами, результат запишите в новую строку. Функция должна иметь прототип:

`char* strSwapParts(const char* str),`

где `str` – строка-источник.

12. Написать функцию, возвращающую строку с удаленным вхождением символов, чьи индексы кратны натуральному `k`. Функция должна иметь прототип:

`char* strdK(const char* str, unsigned int k),`

где `str` – строка-источник, k – кратность индекса.

13. Написать функцию, возвращающую строку с удаленным каждым вхождением заданного символа. Функция должна иметь прототип:

`char* strdchar(const char* str, const char c),`

где `str` – строка-источник, `c` – удаляемый символ.

14. Написать функцию, вставляющую подстроку в строку, начиная с заданной позиции. Функция должна иметь прототип:

`void strins(char* str, const char* substr, unsigned i),`

где `str` – строка-источник, `substr` – вставляемая подстрока, `i` – позиция вставки подстроки.

15. Написать функцию, возвращающую строку в верхнем или нижнем регистре (для латиницы и кириллицы). Функция должна иметь прототип:

`char* strMkUorL(char* str, bool r),`

где `str` – строка-источник, `r` – признак регистра (например, `r = true` – верхний регистр, `r = false` – нижний).

16. Написать функцию, преобразующую в строке все строчные буквы (как латинские, так и кириллицы) в прописные, а прописные — в строчные. Функция должна иметь прототип:

`char* strChngReg(char* str),`

где `str` – строка-источник.

17. Написать функцию, возвращающую строку с удаленным каждым вхождением заданной подстроки. Функция должна иметь прототип:

`char* strdstr(char* str, const char* substr),`

где `str` – строка-источник, `substr` – удаляемая подстрока.

18. Написать функцию, возвращающую подстроку, начинающуюся с заданной позиции и заканчивающейся ближайшим справа от индекса заданным символом исходной строки или, если справа от индекса такого символа нет - концом строки. Функция должна иметь прототип:

`char* substr(const char* str, unsigned i, const char c),`

где `str` – строка-источник, `i` – индекс начала подстроки, `c` – искомый символ.

19. Написать функцию, удаляющую из строки-источника каждое вхождение заданной подстроки начиная с заданной позиции. Функция должна иметь прототип:

`void strdstri(char* str, const char* substr, unsigned i),`

где `str` – строка-источник, `substr` – удаляемая подстрока, `i` – позиция, с которой необходимо начинать поиск для удаления.

20. Написать функцию, вычисляющую количество вхождений заданного символа в строку-источник, начиная с заданной позиции. Функция должна иметь прототип:

`unsigned strccount(char* str, const char c, unsigned i),`

где `str` – строка-источник, `c` – искомый символ, `i` – заданная позиция начала счета.

21. Написать функцию, формирующую строку из исходной, используя шифр Цезаря (каждая буква заменяется на следующую по алфавиту через K позиций по кругу). Функция должна иметь прототип:

`char* str(char* str, int k),`

где `str` – строка-источник, `k` – код шифра.

22. Написать функцию, вычисляющую количество вхождений заданной подстроки в строку-источник. Функция должна иметь прототип:

`unsigned strscount(char* str, char* substr),`

где `str` – строка-источник, `substr` – искомая подстрока.

23. Написать функцию, заменяющую каждое вхождение заданной подстроки в строке-источнике, на другую заданную подстроку. Функция должна иметь прототип:

`void strreplace(char* str, const char* substr1, const char* substr2),`

где `str` – строка-источник, `substr1` – заменяемая подстрока, `substr2` – шаблон замены.

24. Написать функцию, изменяющую длину строки-источника до заданной длины строки. Если заданная длина строки меньше длины строки источника, то строка-источник усекается, если заданная длина строки больше длины источника, то строка источник дополняется пробелами в конце строки. Функция должна иметь прототип:

`void strsetl(char* str, unsigned i),`

где `str` – строка-источник, `i` – заданная длина строки.

25. Написать функцию, выполняющую обмен значениями между двумя заданными строками (аналог функции `swap`). В случае, если строки имеют разную длину, выполнить перераспределение памяти под строки. Функция должна иметь прототип:

`void strswap(char** str1, char** str2),`

где `str1` – адрес 1-й строки, `substr` – адрес 2-й строки.

26. Написать функцию, выполняющую преобразование строки в целое число (со знаком). Функция должна иметь прототип:

`long strtoint(char* str),`

где `str` – строка, содержащая символьное представление целого числа со знаком.

При решении задачи недопустимо использовать существующие в языке функции преобразования строки в число.

27. Написать функцию, выполняющую преобразование строки в действительно число (со знаком). Функция должна иметь прототип:

```
double strtofloat(char* str),
```

где `str` – строка, содержащая символьное представление действительного числа со знаком.

При решении задачи недопустимо использовать существующие в языке функции преобразования строки в число.

28. Написать функцию, выполняющую преобразование целого числа (со знаком) в строку. Функция должна иметь прототип:

```
char* inttostr(long value),
```

где `value` – значение целого числа со знаком.

При решении задачи недопустимо использовать существующие в языке функции преобразования числа в строку.

29. Написать функцию, выполняющую преобразование действительного числа (со знаком) в строку. Функция должна иметь прототип:

```
char* floattostr(double value),
```

где `value` – значение действительного числа со знаком.

При решении задачи недопустимо использовать существующие в языке функции преобразования числа в строку.

30. Написать функцию, вставляющую заданную подстроку в строку-источник в каждую из позиций, заданных массивом индексов. Функция должна иметь прототип:

```
void strinsi(char* str, char* substr, unsigned* i),
```

где `str` – строка-источник, `substr` – строка-вставки, `i` – массив позиций вставки.

31. Написать функцию, выполняющую сортировку символов в заданной строке (по возрастанию или убыванию кодов ASCII). Функция должна иметь прототип:

```
void strsort(char* str, bool f),
```

где `str` – строка-источник, `f` – признак сортировки, например `f = true` – сортировка по возрастанию, `f = false` – сортировка по убыванию.

32. Написать функцию, формирующую строку из двух заданных, состоящую из символов присутствующих в обеих строках. Функция не должна изменять исходные строки. Функция должна иметь прототип:

```
char* strintrsct(const char* str1, const char* str2),
```

где `str1` – 1-я заданная строка, `str2` – 2-я заданная строка.

33. Написать функцию, формирующую строку, состоящую из всех неповторяющихся символов заданной строки (например, для строки «abcad fbe» результат должен быть «cdfe»). Функция должна иметь прототип:

```
char* strdiff(char* str),
```

где `str` – заданная строка.

34. Написать функцию, формирующую строку, состоящую из всех различных символов заданной строки (например, для строки «abcad fbe» результат должен быть «abcdfe»). Функция должна иметь прототип:

```
char* strdiffc(char* str),
```

где `str` – заданная строка.

35. Напишите функцию, которая проверяет, можно ли получить из одной строки другую путем перестановки ее символов. При этом регистром букв нужно пренебречь (например, TomMarvoloRiddle и IamLordVoldemort). Функция должна иметь прототип:

```
bool strIsPerm(const char* str1, const char* str2)
```