ЛАБОРАТОРНАЯ РАБОТА 5
КАКИЕ ТЕМЫ ЗАТРОНУТЫ

- указатели;
- динамические массивы;
- работа с текстом через массивы типа **char**.

ТРЕБОВАНИЯ К ПРОГРАММАМ

- все требования из лабораторных №1, №2, №3, №4. Вновь обратить внимание на выделение смысловых блоков кода в отдельные функции;
- под «предложением» понимается группа любых текстовых символов, включая пробелы, до первого некоторого заданного символа. Как пример, перенос строки — '$\backslash n$' или точка — '.';
- под «словом» понимается группа непробельных символов;
- работу со строками реализуем **только** через динамические массивы типа **char**;
- в заданиях предполагается по умолчанию работа с английским алфавитом;
- словосочетания «ввести строку», «ввести предложение», «задана строка» или «задано предложение» означают два сценария получения информации с консоли:
  1. **ввод строки ограниченной длины**: у пользователя запрашивается значение длины строки, задаётся динамический массив нужного размера и затем вводится сама строка с учётом ограничения на размер. В данном случае — помнить о важности *символа конца строки*;
  2. **ввод строки до определённого символа**: вами выбирается некоторый символ, который будет означать окончание вводимой строки. Как написано выше: точка, точка с запятой, символ переноса строки — что хотите. И программа динамически подстраивает размер массива так, чтобы можно было сохранить все текстовые фрагменты до указанного символа;
- **строго обязательно** корректное возвращение динамической памяти в ОС. Другими словами, на каждый вызов оператора **new[]** должен приходиться вызов оператора **delete[]**. Даже если память выделяется внутри функции **main** — оправдания в стиле «после завершения программы всё само вернётся» не принимаются.

ВСПОМОГАТЕЛЬНЫЕ ФРАГМЕНТЫ

1. В некоторых заданиях встречается понятие **палиндрома** (относительно текста). Это текстовый фрагмент (слово или предложение), который читается одинаково как слева-направо, так и справа-налево. Для отдельных

слов игнорируется регистр символов; для предложений — ещё и знаки пунктуации и пробелы.

Пример английских слов-палиндромов: «madam», «civic», «radar», «racecar», «level».

Пример предложений на английском: «Mr. Owl ate my metal worm», «step on no pets», «was it a car or a cat I saw?».

ЗАДАНИЯ

5.1. Ввести строку текста и найти слова, в которых количество согласных букв больше, чем гласных. Под гласными понимать только буквы английского алфавита: «**a e i o u y**», под согласными — любые другие символы.

5.2. Ввести строку текста и найти слова, которые состоят из чётного количества символов.

5.3. Ввести предложение и найти в нём слова, которые содержат в себе цифры.

5.4. Ввести предложение и найти в нём слова, которые содержат **повторяющиеся подряд** буквы (под «буквой» понимаются символы английского алфавита в нижнем или верхнем регистрах).

5.5. Ввести предложение и найти в нём самое длинное слово.

5.6. Ввести строку. В ней заменить все повторяющиеся подряд символы в словах на фрагмент, состоящий из самого символа и количества его повторений. Пример: «cooooouuunnnnnntt» => «co4u3n6t2»

5.7. Ввести n предложений. Среди введённых найти те, которые содержат в себе *слова-палиндромы*. Ввод предложений должен быть произведён отдельно от их обработки.

5.8. Ввести два предложения. Выяснить, какое из них содержит в себе больше целых положительных чисел. Под *целым числом* в предложении понимать только отдельную группу цифровых символов, которая отделена от других слов пробелами.

5.9. Заданы две строки. Создать новую строку, которая состоит из символов, которые входят в обе заданные строки.

5.10. Заданы две строки. Создать новую строку, которая состоит из символов, которые присутствуют в первой строке, но не присутствуют во второй.

5.11. Задано предложение и положительное целое число. Найти все слова, в которых число символов больше заданного числа.

5.12. Задано предложение и положительное целое число. Найти все слова, в которых число символов меньше заданного числа.

5.13. Ввести предложение и два слова. Заменить в предложении все вхождения первого слова на второе. Если ни одного вхождения не найдено — вывести информативное сообщение.

5.14. Реализовать функцию по удалению из строки «лишних» пробелов. Лишними считаются:

1. все пробелы от начала строки до первого непробельного символа;
2. все пробелы после последнего непробельного символа и до конца строки;
3. все повторяющиеся пробелы между словами строки (для примера, из трёх пробелов между словами остаётся только один);

Продемонстрировать работу функции при задании n строк, n — определяется пользователем.

5.15. Ввести два предложения. Поменять местами **чётные** слова первого предложения с **нечётными** словами второго. Обмен прекращается, когда заканчиваются слова в одном из предложений.

5.16. Задано предложение и слово-шаблон для поиска. Шаблон зададим следующим образом: это группа символов, состоящая из цифр, букв английского алфавита и символа '*****'. '*****' обозначает любой произвольный символ, кроме пробела. Найти во введённом предложении все слова, соответствующие заданному шаблону. Для примера, есть предложение «We have enough skills to complete programming languages course successfully», то шаблон «*e» (любой символ, следующий перед буквой 'e') выберет слова «We», «have», «complete», «languages», «course», «successfully», шаблон «ill» — только «skills», а шаблон «*ll*» — «skills» и «successfully».

5.17. Ввести n предложений и некоторое слово. Найти все предложения, в которых встречается заданное слово и отсортировать в порядке убывания частоты встречи (т.е. сначала идёт предложение, в котором слово встретилось наибольшее количество раз и далее по убыванию).

5.18. Ввести n предложений. Найти все из них, которые являются палиндромом (в смысле полного предложения, а не отдельных слов).

5.19. Реализовать простой шифратор/дешифратор строки. Шифрование происходит по принципу: от каждого символа английского алфавита смещаемся «вправо» на заданное количество позиций и заменяем текущий символ на получаемый после смещения. Для работы с произвольным смещением, алфавит условно «закольцовываем»: после буквы 'z' идёт 'a', после 'Z' => 'A'. Небуквенные символы предложения остаются без изменений. Размер смещения задаётся пользователем. Продемонстрировать шифровку и расшифровку введённого предложения.

5.20. Задана строка, в которой все слова записаны в обратном порядке. Восстановить правильный порядок каждого слова.

5.21. Задано предложение и некоторое слово. Выбрать все слова из предложения, которые можно составить, используя только буквы введённого слова. Для примера, если введено слово «windows», то в предложении можно выбрать слова «win», «wins», «now», «down», «downs» (если, конечно, они присутствуют в нём).

5.22. Заданы две строки. Удалить из первой строки все **повторяющиеся** слова, которые присутствуют во второй строке (другими словами, слово удаляется из первой строки, если оно встречается в ней более одного раза и одновременно присутствует во второй строке).

5.23. Задана строка. Каждое её слово преобразовать по следующему правилу: все символы слова, равные последнему символу, заменяются на заданный пользователем символ. Для примера, есть слово «execute» и символ '@'. После преобразования получаем слово «@x@cute».

5.24. Заданы две строки. Если первая строка является строкой-палиндромом, то во втором предложении заменить все буквы от 'a(A)' до 'h(H)' включительно на их коды из кодировки ASCII.

5.25. Ввести предложение и два слова. Заменить в предложении все вхождения первого слова на второе с указанием номера замены (первая замена, вторая, третья и так далее). Если ни одного вхождения не найдено — вывести информативное сообщение.

5.26. Задано предложение и некоторое слово. Выбрать все слова из предложения, в которых не присутствует ни один символ введённого слова.

5.27. Ввести n предложений. В каждом из них переставить слова в обратном порядке.

5.28. Ввести n предложений. В каждом из них поменять местами слова с наибольшей и наименьшей длиной.

5.29. Заданы два слова. С их помощью составить все возможные варианты кроссвордов. Для примера, даны слова «phenomena» и «place». Тогда возможные варианты:

1.

$p\ h\ e\ n\ o\ m\ e\ n\ a$
l
a
c
e

2.

p
l
a
c
$p\ h\ e\ n\ o\ m\ e\ n\ a$

3.

p
l
a
c
$p\ h\ e\ n\ o\ m\ e\ n\ a$

4.

p
l
$p\ h\ e\ n\ o\ m\ e\ n\ a$
c
e

5.30. Треугольником Паскаля называют специальную матрицу, каждый элемент которой рассчитывается по формуле

$$a_{nk} = \frac{n!}{k!(n-k)!},$$

где n обозначает номер строки, k — номер столбца. Нумерация n начинается с нуля, k меняется от нуля до n включительно. Первые четыре строки матрицы выглядят так (в первой строке — один элемент, во второй — два, в третьей — три и т.д):

$n \setminus k$	0	1	2	3	4
0	1				
1	1	1			
2	1	2	1		
3	1	3	3	1	
4	1	4	6	4	1

С помощью этого треугольника можно придумать шифрование для строки текста. Принцип следующий: берём слово из текста; его длина определяет номер строки треугольника Паскаля; каждая английская буква слова определяет номер столбца. Небуквенные символы пропускаем. Каждую английскую букву в слове «сдвигаем в алфавите влево» на число позиций, заданное значением элемента в треугольнике Паскаля. Для примера, слово «cat» имеет три буквы, это строка с $n = 2$ выше. Тогда, буква 'c' заменяется на 'b' (смещение на один символ влево), буква 'a' на 'y' (закольцовываем алфавит), буква 't' на 's'. Итого «cat» преобразовано в «bys».

Ввести строку и показать на её примере работу шифратора и дешифратора.

5.31. Реализовать функцию для разбиения строки на отдельные фрагменты по заданному разделителю. Разделитель сам по себе может быть произвольным набором символов. Так, если разделитель — пробел, то осуществляем деление по словам.

Ввести строку и разделитель. Вывести на консоль массив фрагментов после разбиения.

5.32. Дано n предложений. Вывести в алфавитном порядке все встречающиеся в них слова. Каждое слово выводить по одному разу.

5.33. В заданном предложении найти все анаграммы. Одно слово является анаграммой другого, если оно может быть получено перестановкой букв данного слова. Для примера, анаграммами являются следующие группы слов: {«float», «aloft»}; {«cartesian», «ascertain»}; {«verbose», «observe», «obverse»}.

5.34. Дано предложение с набором ключевых слов и еще n предложений. Вывести предложения в порядке убывания количества входящих в них ключевых слов. Предложения, в которые не входит ни одного ключевого слова, не выводить.

5.35. Дано два предложения. Определить, входят ли все слова из второго предложения в первое. Слова могут идти не подряд, но порядок вхождения должен сохраняться.

5.36. Дано предложение. Заменить в нем все целые числа на соответствующую им словесную запись. Под *целым числом* в предложении понимать только отдельную группу из не более 9 цифровых символов, которая отделена от других слов пробелами. Для примера, предложение «The speed of light equals to 299792458 m/s» должно быть преобразовано в предложение «The speed of light equals to two hundred and ninety-nine million, seven hundred and ninety-two thousand, four hundred and fifty-eight m/s».