Блок 3. Машины (обязательные) по теме «Векторы»

Требование к сдаче программ, начиная с 3-го блока:
Во всех задачах верхние границы массивов объявлять **константами с указанными ниже** (в условиях задач) **значениями**, иначе проверка задач производиться не будет.

8.29ж (n=10) Программа вводит элементы исходного массива (*1-ый этап*), преобразует этот массив (*2-ой этап*), и распечатывает его новое содержимое (*3-ий этап*). *Смешивать этапы нельзя*! В решении запрещено использовать вспомогательный массив; нельзя использовать вложенные циклы; требуемое преобразование следует сделать за один просмотр массива. Внимание: в результате преобразований ненулевые элементы должны быть расположены в исходном порядке их вхождения в массив.

Подсказка для этой задачи (решать только согласно подсказке):
Используем два указателя `i` и `j` с начальной установкой: `i:=1; j:=1;`
Смысл `i` – настроен на текущую проверяемую позицию (меняется от `1` до `n`).
Смысл `j` – настроен на позицию, в которую будет пересылаться очередной ненулевой элемент (после пересылки настраиваемся на следующую позицию: `j:=j+1`).
Получается, что если в начале массива стоят ненули, то эти ненули как бы переписываются на свои прежние позиции (т.е. значения `i` и `j` на этапе обработки этой части массива будут совпадать). Но как только появится первый ноль, то значение `i` станет опережать значение `j`.
После обработки всех элементов имеем: в начале массива окажутся все ненули. Остается искусственно заполнить хвост массива нулями (от `j`-ой позиции до `n`-ой). Получим желаемое.

8.41 б (n=10) Обратить внимание на замечание (*в скобках*) к этому пункту!

8.41 в (n=10)

8.41 г (n=10) (перед написанием программы следует разобрать задачи **8.36**, **8.38**, **8.40)**

8.51 (n=10)

8.53 (n=10) Организовать двойной цикл для перебора всевозможных пар точек; *требование*: каждая пара должна быть рассмотрена только 1 раз; т.е. нужно использовать цикл вида
`for i:=1 to n-1 do for j:=i+1 to n do …`

8.54 (n=10) За начальное значение для минимума взять MaxInt; для каждой последовательности завести свой массив; во внешнем цикле с заголовком `for i:=1 to n do` перебираем все элементы первого массива, и для каждого такого элемента запускаем вспомогательный цикл по поиску совпадающего элемента во втором массиве; *требование*: при решении задачи не делать лишних сравнений, т.е. досрочно прерывать внутренний цикл при первом же совпадении

8.55 (n=10) Нужен двойной цикл для перебора всевозможных пар чисел, т.е. цикл вида
`for i:=1 to n-1 do for j:=i+1 to n do …` во внешнем цикле - очередной левый элемент из пары, а во внутреннем – всевозможные элементы справа от него

8.56 Завести вспомогательный массив с индексами от `'a'` до `'z'` для хранения числа вхождений каждой латинской буквы

8.58 (n=10) Запрещено решать задачу исходя из знаний размера используемой кодовой таблицы, т.е надо решать по мотивам семинарских задач **8.35** и **8.57 (вспомогательных массивов в решении не использовать!)**

Итого: 10 обязательных задач. Срок сдачи: до 1 ноября включительно.

см. дополнительные задачи на следующей странице

Блок 3. Машины (дополнительные) по теме «Векторы»

8.59 (5 очков)
Примеры работы программы:
1) `sas,aaas,saaa,sb,bbbbb,b. → s`
2) `qw,azz,qwwwq,wer,ert,edf,ok,aaaaa. → e w`

Подсказка для этой задачи: завести два вспомогательных массива со следующим описанием
```
var
    L: array['a'..'z'] of integer;
```
{накапливает число вхождений букв в словах последовательности}
```
    W: array['a'..'z'] of boolean;
```
{рабочий массив, хранит для каждого слова информацию о буквах: какие из них входят в слово (этот массив нужен, чтобы повторяющуюся в некотором слове букву не учитывать несколько раз в массиве L)}

8.39 (5 очков) (размерность массива оформить как константу **n=10**)
Программа запрашивает пользователя задать (=ввести с клавиатуры) число (от **1** до **11**), которого не будет в массиве. Затем программа генерирует массив нужного вида и распечатывает (построчно) его элементы. Далее программа, в предположении, что отсутствующий элемент неизвестен, находит его **методом бинарного поиска** (см. **8.38** из задачника) и выводит найденный ответ на экран. Оба числа (заданный пользователем отсутствующий элемент и найденный методом бинарного поиска элемент) при правильном решении должны совпасть.

Доп_задача_3 «Разделение массива» (10 очков)

```
const   n = 15;
var  a:  array[1..n] of integer;  x: integer;
```

Ввести **n** целых чисел и сохранить их в массиве **a** размерности **n**. Затем ввести целое число **x** (оно не обязано совпадать с каким-либо элементом массива).
Необходимо переставить элементы массива **a** так, чтобы сначала в нём следовали (в произвольном порядке) элементы, меньшие (<) числа **x**, затем - элементы, равные (=) числу **x** (если такие есть), а затем – элементы, большие (>) числа **x**.

Требование: решить задачу не более чем за два прохода по массиву (дополнительных массивов не использовать).

Подсказка.
1 этап. Просматриваем элементы массива **x** слева направо, пока не встретим элемент **a[i]>=x**.
Затем просматриваем элементы массива справа налево, пока не встретим элемент **a[j]<x**.
Найденные элементы меняем местами и продолжаем процесс со следующих элементов.
Конец 1-го этапа: массив разделён на две части. 1_часть: все элементы, меньшие **x**, 2_часть: все элементы, большие или равные **x**.
2 этап: **i** стоит на начале 2-ой части, **j** установим на конец массива. Пропускаем (двигаясь слева направо) все элементы **x[i]**, равные **x**. Пропускаем (двигаясь справа налево) все элементы **x[j]**, отличные от **x**. Меняем элементы местами и продолжаем движение со следующих элементов.