Машины-2 (массивы – обязательные задачи)

(все девять задач - **обязательные**, их нужно **сдать до 4 апреля**)

Задача 1 . “ Первые вхождения ”

Ввести (посимвольно, т.е. с использованием макрокоманды **inchar**) текст из больших латинских букв, заканчивающийся **точкой**, и вывести его (**без точки**), сохранив в нем только **первые вхождения** каждой буквы. *Замечание:* в используемой нами кодировке большие латинские буквы упорядочены согласно алфавиту и следуют друг за другом без пропусков.

Пример: **BABCDBADEBFA.** Будет напечатано: **BACDEF**

Подсказка: Для решения задачи предлагается описать в программе вспомогательный байтовый массив **LAT**:

```
LAT db 'Z'-'A'+1 dup(0); LAT['A'..'Z'] of 0..1
```

;LAT[i] соответствует **i**-ой букве латинского алфавита (при **внутренней** нумерации букв/элементов от нуля)

В процессе решения задачи этот массив следует преобразовывать по следующему правилу:

LAT[i]=
- **1,** если **i**-ая буква уже встречалась в тексте и была напечатана
- **0,** если **i**-ая буква ещё не встречалась в тексте

В начальный момент не прочитано (т.е. не напечатано) ни одной буквы, поэтому элементы массива инициализированы нулями (при распределении памяти под массив с помощью директивы **db**).

Рекомендации. см. решение домашней задачи **6.28** (решение было отправлено *23.03.2021* по почте в виде готовой программы). Решать по аналогии с этой задачей.

Задача 2 . “ Зачёт с оценкой ”

На вход в программу поступает (по макрокоманде **inint**) целое число от **2** до **5** (ввод корректный). Требуется напечатать одно из следующих слов: **неуд** (при вводе числа **2**), **удовл** (при вводе числа **3**), **хорошо** (при вводе числа **4**), **отлично** (при вводе числа **5**).
Требование: **команды переходов и циклов в решении использовать запрещено**.

Подсказка: Описать в программе следующие символьные строки:

```
z2 db 'неуд',0
z3 db 'удовл',0
z4 db 'хорошо',0
z5 db 'отлично',0
```

; *внимание*: целочисленный **ноль** в конце каждой строки нужен для работы макрокоманды **outstr[ln]**, чтобы макрокоманда знала, где остановить вывод

Описать также массив, элементами которого являются **адреса** этих строк:

adr dd z2,z3,z4,z5; адреса – всегда 32-битовые !

Рекомендации. После считывания числа (от **2** до **5**) следует определить индекс соответствующего элемента массива **adr** (от **0** до **3**) – вычитанием из числа двойки (и поместить этот индекс в какой-нибудь модификатор, например, в **EBX**). Далее умноженное на 4 (т.к. элементы массива **adr** – типа **dword**) значение найденного индекса использовать для доступа к **адресу** нужной строки (этот **адрес** – элемент массива **adr**). Поместить найденный адрес на какой-нибудь регистр, например, на тот же **EBX**, после чего воспользоваться макрокомандой **outstr EBX** для вывода ответа.

Требование: для умножения на 4 не надо пользоваться командой **MUL** (а надо воспользоваться **масштабным множителем 4** при доступе к элементу массива **adr**)
Замечание (для внимательных студентов): полученное решение можно усовершенствовать, если корректировку исходного числа (2..5) перенести на последний этап (при доступе к элементу массива **adr** можно подкорректировать его адрес вычитанием нужного числа байт).

Задача 3 . " Картинка"

Ввести по макрокоманде **inint ECX** целое число К из диапазона **[1..20].** Используя только макрокоманды **outstr[ln] и newline**, напечатать картинку из К строк.
Каждая строка состоит из К символов: в начале строки идут минусы, а затем – звёздочки.
Для К=4 должна получиться картинка вида (аналогично для других значений К):

```
- - - *
- - * *
- * * *
* * * *
```

Требования: 1) программа должна содержать не более одного цикла; 2) использование макрокоманды **outchar** в решении запрещено.
Рекомендации. Описать в программе вспомогательный массив:
S db 20 dup('-'),0; `S[0..20], где S[20]=0`
До входа в основной цикл обнулить байт с индексом **К** в массиве **S** (для возможности использовать макрокоманду вывода строки длиной **К** символов). Далее, на каждом шаге цикла видоизменять (сообразить как) содержимое массива **S** так, чтобы выводить (на этом шаге цикла) по **outstr[ln]** строку требуемого вида.

Задача 4 . " Знаковое 10-е число"

Ввести по макрокоманде **inint EAX** целое число из диапазона **[-2^{31}..2^{32}-1]** (этот диапазон является объединением диапазона знаковых и беззнаковых *32*-битных чисел). Используя только макрокоманды **outchar и outstr[ln]**, вывести содержимое регистра **EAX** в виде **знакового десятичного числа**.

Рекомендации. Сначала нужно выяснить знак числа, попавшего на регистр **EAX**, и вывести (по **outchar**) символ **'-'**, если число отрицательное. Далее нужно работать с **абсолютной величиной числа**.

Для формирования символьного представления абсолютной величины числа предлагается использовать **вспомогательный 10-байтный массив** (так как для вывода *32*-битных величин требуется не более 10 десятичных цифр). Заполнять этот массив предстоит от конца к началу (т.к. сначала найдём младшие цифры числа, а затем – более старшие). **11**-ым байтом вслед за массивом обязательно должен быть **целочисленный ноль** для возможности вывода ответа по макрокоманде **outstr[ln]** (разрешается вывод незначащих нулей, но **лучше их не печатать**). Настроить какой-нибудь регистр, например, **EDX** на начало (или на первую значащую цифру) этого символьного массива (чтобы вывести с помощью макрокоманды **outstr[ln]** последовательность найденных десятичных цифр числа).

Искомые ***10***-е цифры формировать методом деления на **10** и взятия остатков; при делении учесть, что неполное частное (**div**) может не уместиться в *8*-битовый или *16*-битовый форматы (т.е. здесь потребуется реализовать "сверхдлинное" деление).

Чтобы не выводить незначащие нули (это дополнительная опция, но весьма желательная), полезно воспользоваться командой **loopne** (при **ECX=10**), выполняя проверку на равенство очередного **div** нулю последней командой тела цикла (см. об этой команде в *примечании* к заданию). Тогда после выхода из цикла значение в **ECX** будет соответствовать количеству незаполненных начальных элементов массива (а значит будет ясно, на какую величину надо подкорректировать значение в **EDX** перед выводом ответа по макрокоманде **outstr[ln]**).

Примечание. Действие команды **loopne L**:
`ECX:=ECX-1; if (ECX<>0)and(ZF=0) then goto L`

Задача 5 . “ Беззнаковое 16-е число”

Ввести по макрокоманде **`inint EAX`** целое число из диапазона **`[-2^31..2^32-1]`** (этот диапазон является объединением диапазона знаковых и беззнаковых 32-битных чисел). Используя только макрокоманду **`outstr[ln]`**, вывести содержимое регистра **`EAX`** в виде **беззнакового шестнадцатеричного числа**.
Требование: обязательно выводить все восемь шестнадцатеричных цифр (включая незначащие нули) !

Рекомендации. Идея решения аналогична идее, предложенной в **Задаче 3**, но делить теперь надо на **`16`**. При преобразовании числовой величины (от **`0`** до **`15`**) в соответствующую 16-ричную цифру (от **`'0'`** до **`'F'`**) различать два случая: **1)** число от **`0`** до **`9`** (преобразуется в символ от **`'0'`** до **`'9'`**); **2)** число от **`10`** до **`15`** (преобразуется в символ от **`'A'`** до **`'Z'`**) (см. решённую на семинаре задачу **5.15**).

Задача 6 . “ Сортировка выбором”

```
N equ 30          ; предельный размер рабочего массива для хранения введённых элементов
X dd N dup(?)     ; числа со знаком
```

Программа запрашивает фактическую размерность массива (в решении можно рассчитывать на то, что размерность будет задаваться числом от 5 до 30). Далее пользователь вводит элементы массива (числа со знаком). Программа упорядочивает массив X по неубыванию ($X_1 \le X_2 \le X_3 \le ...$), используя следующий метод сортировки: *найти максимальный элемент массива и переставить его с последним элементом; затем этот же метод применить ко всем элементам, кроме последнего (он уже находится на своем окончательном месте); и т.д.* Отсортированный массив выводится на экран (продумать вывод, чтобы выводимое число не оказалось в конце одной и в начале другой строки).

Рекомендация. Память под массив выделить с расчетом на максимально возможное значение N (=30), а далее работать с начальной частью массива – с учётом его фактической длины.

Задача 7 . “ Сортировка обменом (метод пузырька)”

```
N equ 30          ;  предельный размер рабочего массива для хранения введённых элементов
X dd N dup(?)     ;  числа без знака
```

Программа запрашивает фактическую размерность массива (в решении можно рассчитывать на то, что размерность будет задаваться числом от 5 до 30). Далее пользователь вводит элементы массива (числа без знака). Программа упорядочивает массив X по неубыванию ($X_1 \le X_2 \le X_3 \le ...$), используя следующий метод сортировки: *последовательно сравнивать пары соседних элементов массива (X_1 с X_2, X_2 с X_3 и т.д.) и, если первый элемент пары больше второго, то выполнить перестановку пары; тем самым наибольший элемент окажется в конце массива; затем этот же метод применить ко всем элементам, кроме последнего; и т.д.* Отсортированный массив выводится на экран (продумать вывод, чтобы выводимое число не оказалось в конце одной и в начале другой строки). *Требование*: *если при очередном просмотре массива не было сделано ни одной перестановки, то на этом прекратить работу в цикле (т.к массив уже отсортирован)*.

Рекомендация. Память под массив выделить с расчетом на максимально возможное значение N (=30), а далее работать с начальной частью массива – с учётом его фактической длины.

Задача 8 . “ Палиндром ?”

```
N equ 30  ;  предельный размер рабочего массива для хранения введённых символов
S db N dup(?)   ;  S[0..N-1] – рабочий массив
```

Ввести *текст* из не более **N** *латинских букв* и *пробелов* (конец последовательности – точка). Определить, является ли этот *текст* палиндромом (т.е. “перевёртышем”) при следующих условиях: 1) пробелы должны быть проигнорированы; 2) малые и большие латинские буквы – отождествляются (т.е. не различаются).
Например:
`_ _ S_ _ adf _ _ F _ _ d_ _ aS _ _` **.** → **палиндром** (здесь **_** обозначает пробел)

Ответ в виде: **палиндром / не палиндром** (использовать решённую на семинаре задачу **6.22_a**)

Задача 9 . “ Минимальный элемент матрицы”

```
N equ 4                  ; количество строк матрицы
M equ 6                  ; количество столбцов матрицы
A dd N dup(M dup(?)) ; матрица A[1..N,1..M] 32-битовых чисел со знаком
```

Ввести (по **inint**) элементы матрицы **A**.
Вывести на экран минимальный элемент матрицы, число его вхождений и индексы всех вхождений этого минимального элемента. *Например*, при вводе матрицы:

1	2	3	4	-500	- 2000000000
-2000000000	-1000000000	-2000000000	2000000000	5	0
1	2	3	4	-2000000000	-500
-1000000000	-2000000000	-3	-4	-5	-2000000000

должно быть напечатано:
минимальный элемент **-2000000000** входит **6** раз
индексы его вхождений: [1,6] [2,1] [2,3] [3,5] [4,2] [4,6]

Внимание! Учесть, что по смыслу задачи элементы строк и столбцов матрицы нумеруются от единицы.

Рекомендация. Вводить элементы матрицы однократным циклом, рассматривая матрицу как одномерный массив размерности N*M. На этапе ввода определить *минимальный элемент* и *число его вхождений*.
На втором этапе запустить двойной цикл (внешний – по строкам, внутренний – по элементам текущей строки) для определения индексов всех вхождений найденного минимального элемента. Не забыть, что с точки зрения пользователя строки и столбцы нумеруются от единицы (а при представлении в памяти – нумеруются от нуля) – ответ должен быть выдан с точки зрения пользователя.