

Машины-2 (массивы – обязательные задачи)

(все шесть задач - **обязательные**, их нужно сдать до **8 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 26 dup(0)
;LAT[i] соответствует i-ой букве латинского алфавита (при нумерации букв от нуля)
```

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

$$\text{LAT}[i] = \begin{cases} 1, & \text{если } i\text{-ая буква (по алфавиту) уже встречалась в тексте и была напечатана} \\ 0, & \text{если } i\text{-ая буква (по алфавиту) ещё не встречалась в тексте} \end{cases}$$

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

Рекомендации. После считывания очередной буквы следует определить её порядковый номер в английском алфавите (при нумерации букв от нуля, т.е. номер 'A' – 0, номер 'B' – 1, номер 'C' – 2 и т.п.) путем вычитания кода буквы 'A' из кода прочитанной буквы. Тем самым получим индекс нужного нам элемента массива **LAT**. Чтобы обратиться к элементу с этим индексом, помещаем найденный индекс в какой-нибудь модификатор, например, **EBX**, и далее используем конструкцию **LAT[EBX]** для доступа к этому элементу. Если в соответствующем элементе хранится 0, то записываем в него 1, после чего печатаем прочитанную букву. Если в элементе хранится 1, то букву не печатаем (т.к. она была напечатана ранее).

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 продвинутых 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. “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 записи 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 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 4. “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 **4.7**).

Задача 5. “Сортировка выбором”

N EQU 30

X DD N DUP(?) ; числа со знаком

Программа запрашивает 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 ($X_1 \leq X_2 \leq X_3 \leq \dots$), 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 6. “Сортировка обменом (метод пузырька)”

$N \text{ EQU } 30$

$X \text{ DD } N \text{ DUP}(?)$; числа без знака

Программа запрашивает 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 ($X_1 \leq X_2 \leq X_3 \leq \dots$), 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 переставлять их; 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.