

Задача**«Ввод-вывод четверичного числа»**

В задаче рассматривается работа с упакованными четверичными числами.

Для представления таких чисел будем использовать двойные слова (занимающие в памяти, как известно, 32 разряда). Каждая цифра четверичного числа в этом случае кодируется парой соответствующих двоичных цифр: $0 \rightarrow 00$, $1 \rightarrow 01$, $2 \rightarrow 10$, $3 \rightarrow 11$. Например, 32-битное двоичное число **00001011010000110110001110100001** является упакованным представлением четверичного числа **23100312032201** (незначащие двоичные нули в этой записи отброшены). Заметим, что это же 32-битное двоичное число (**00001011010000110110001110100001**) является также внутренним машинным представлением беззнакового десятичного числа **188965793**.

Формулировка задания.

Написать программу из *трёх модулей*.

В *головном модуле* описать 32-битную переменную с именем **X** (содержимое которой трактуется как упакованное представление некоторого четверичного числа). В *1-ом вспомогательном модуле* описать две **общедоступные** процедуры со стандартными соглашениями о связях `stdcall`: процедуру **In4(X)** и процедуру **Out4(X)** (их имена будут **внешними** для *головного модуля*).

Процедура **In4(X)** – для посимвольного ввода цифр четверичного числа (*конец записи четверичного числа* – **пробел**) и формирования 32-битного упакованного представления введённого числа. Полученное представление возвращается через параметр процедуры (следовательно, параметр необходимо передать **по ссылке**). Считать, что величина вводимого числа укладывается в формат двойного слова.

Процедура **Out4(X)** – для вывода содержимого двойного слова **X** в четверичном виде (цифра за цифрой). Параметр передать в процедуру **по значению**.

Из *головного модуля* сначала вызывается процедура **In4(X)**, с помощью которой вводится четверичное число и присваивается параметру процедуры. Затем из *головного модуля* вызывается процедура **Out4(X)**, по которой введённое число распечатывается в четверичном виде.

Введённое и напечатанное числа должны, естественно, совпасть.

Затем из *головного модуля* управление передаётся (по внешней метке) во *2-ой вспомогательный модуль*, где печатается значение переменной *X* в десятичном виде б/зн (по **outword X**), на этом работа программы завершается (без возврата в головной модуль). Из *2-го вспомогательного модуля* доступ к переменной *X* осуществляется напрямую по её имени (то есть имя *X* во *2-ом вспомогательном модуле* следует объявить как *внешнее*, а в *головном модуле* – как **общедоступное**). Продумать также вопрос о **точке входа** во *2-ой вспомогательный модуль*.

Требование: пользоваться командами умножения и деления при решении задачи – запрещено.

Замечания по реализации процедуры In4(X).

Пусть, например, ответ накапливается на **EAX** по схеме Горнера:

EAX := EAX*4 + (очередная четверичная цифра)

Так как $4=2^2$, то каждый раз достаточно сдвигать влево на 2 позиции содержимое **EAX** и затем складывать (с помощью **or**) младшую часть **EAX** с новой четверичной цифрой. Получим, с одной стороны, упакованное представление введённого четверичного числа, а с другой стороны – 32-битовое двоичное представление введённой беззнаковой величины.

Замечания по реализации процедуры Out4(X).

Для вывода числа в четверичном виде следует поступить так.

Запустить цикл из 16 шагов (максимальное количество цифр в четверичной записи числа). На каждом шаге – сдвигать циклически (**rol**) содержимое **EAX** сразу на 2 разряда влево. В результате – в двух младших разрядах **EAX** окажется пара двоичных цифр, соответствующая очередной четверичной цифре. Выделяем эту цифру, используя маску **11b** (содержимое **EAX** при этом не портим!) и выводим цифру на печать (разрешается выводить незначащие нули).

Требования к оформлению.

Головной модуль назвать **main.asm**

Вспомогательные модули назвать **unit1.asm** и **unit2.asm**

Останов – в модуле **unit2** !

Сдача задания: прислать в архиве три исходных модуля (**main.asm**, **unit1.asm**, **unit2.asm**) и исполняемый файл **main.exe**. Просьба перед **exit** ставить **pause** (я запускаю ваши программы непосредственно из архива, мне важно успеть увидеть результат работы программы).

Плодотворной работы!

И.В.