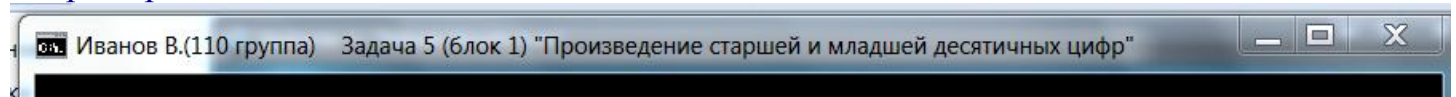


Разобраться предварительно с **вводом-выводом** по пособию В.Г.Баулы (глава 6, раздел 6.5.1), и со **структурой программы на языке ассемблера** (глава 6, раздел 6.5.2). Только после этого приступить к написанию программ.

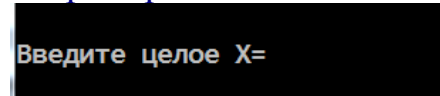
Обязательно усвоить следующие макрокоманды (из раздела 6.5.1): **ClrScr**, **ConsoleTitle**, **outstr[ln]**, **inchar**, **outchar**, **inint[ln]**, **outint[ln]** (или **OutI[ln]**), **outword[ln]** (или **OutU[ln]**), **newline**, **flush**, **pause**, **exit** (обратите при этом внимание, что названия этих макрокоманд – имена пользователя, следовательно, **малые и большие буквы в этих названиях - различаются**). Остальные макрокоманды осваивать не обязательно (но если есть желание и интерес, то их можно использовать, но не увлекаться).

Внимание! Во всех программах выдавать следующую **обязательную информацию в заголовке консольного окна**:

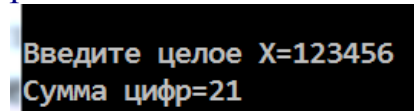
- фамилию и имя студента, номер группы;
- номер решаемой задачи (из какого она блока) и название задачи, если оно имеется, например:



- приглашение к вводу данных (чего ожидаете от пользователя и в каком виде), например:



- объяснение (лаконично), что выдаётся вашей программой в качестве результатов её работы.



Следует комментировать текст программы (лаконично, только наиболее важные этапы и тонкие места).

По каждой задаче присылать три файла - с расширениями **asm**, **lst**, **exe**). Если в архиве несколько задач, то каждую задачу помещайте в отдельную папку с номером этой задачи. Название архива лучше давать с учётом вашей фамилии и номера блока. Можно сдавать по одной задаче или по нескольку (у кого как получится).

Перед макрокомандой **exit** ставьте обязательно макрокоманду **pause** (для удержания экрана с результатами работы программы). **Иначе проверять не будем !!!**

Задачи блока 1 см. на следующей странице.

Задачи (блок 1). Арифметика, логика, циклы.

(Обязательно нужно решить **три любых задачи** из десяти предлагаемых.

За каждую дополнительную задачу начисляется по 10 очков.

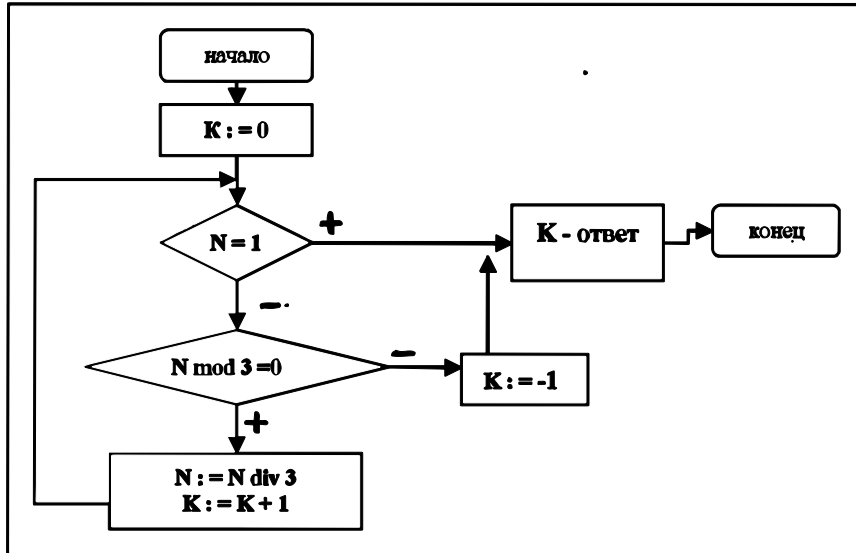
Решения обязательных задач нужно прислать до 1-го апреля, дополнительных – до конца семестра)

Задача 1. “Степень тройки”

Ввести число ≥ 1 (считать, что ввод будет корректный и число укладывается в формат двойного слова). Определить, является ли это число **степенью тройки** (1, 3, 9, 27, ...). Если да, то вывести на экран показатель степени, иначе вывести число **-1**.

Решать согласно следующей **блок-схеме** (условные обозначения: **N** – исходное число, **K** – искомый показатель):

Блок-схема к задаче № 3.12



РЕКОМЕНДАЦИИ: Решать строго *методом деления на 3*: делимое представлять в формате учетверенного слова, а делитель - в формате двойного слова, т.к. нет гарантии, что *результат деления на 3* укладывается в байт или в слово (например, $900/3=300>255$, $300000/3=100000>65535$). То есть требуется сверхдлинное деление. В решении не отклоняться от предлагаемой блок-схемы. Считывать исходное число выгодно сразу на регистр **EAX**; ответ выгодно тоже формировать на регистре (для повышения быстродействия программы). Добейтесь, чтобы в теле цикла не было обращений к ячейкам оперативной памяти.

Задача 2. “Простое число”

$N \geq 2$; $N > 1$

Ввести число **N** (>1), считая, что ввод будет корректный. Проверить, является ли это число простым. Ответ напечатать в виде одного из слов: **ПРОСТОЕ** или **СОСТАВНОЕ**.

РЕКОМЕНДАЦИИ: В цикле делить исходное число на числа от **2** до **$N \div 2$** (с целью поиска первого делителя), т.е. максимальное число шагов цикла **$N \div 2 - 1$** . Как только найдется первый делитель – досрочно выходить из цикла (число составное). Вышли по концу цикла – число простое.

Выбрать правильный вид деления (короткое, длинное или сверхдлинное).

Для облегчения вывода ответа сделать начальное предположение, что число составное (т.е. до начала проверки числа загрузить, например, в регистр **EDX** адрес начала строки со словом “СОСТАВНОЕ”).

Но если в конце выяснится, что делителей у числа не нашлось, то перед выводом ответа (по макрокоманде **outstrln EDX**) скорректировать значение в **EDX** (поместив в него адрес начала строки со словом “ПРОСТОЕ”). **Как поместить адрес начала строки в регистр?** Можно по-разному:

| 1 способ (хороший) | 2 способ (хороший) | 3 способ (не рекомендуется) |
|--|---|--|
| <pre>.data S db “СОСТАВНОЕ”, 0 .code mov EDX, offset S</pre> | <pre>.data S db “СОСТАВНОЕ”, 0 .code lea EDX, S; <i>скоро изучим это!</i></pre> | <pre>.data S db “СОСТАВНОЕ”, 0 adr_S dd S .code mov EDX, adr_S</pre> |

Задача 3. “Баланс скобок”

Ввести последовательность символов (отличных от точки), за которой следует точка (за один сеанс ввода, т.е. набрать всю последовательность символов с точкой на конце и затем нажать ENTER – как это делали в 1-ом семестре). Определить, сбалансирована ли эта последовательность по круглым скобкам. Ответ в виде одного из слов: ДА или НЕТ. (Требование: текст не сохранять в памяти.)

Пример сбалансированной последовательности:

a (b () cd (ef (sdf) k) s) s (s) d.

Примеры несбалансированных последовательностей:

(a) (asd.

(as)) s (sd) .

) (.

РЕКОМЕНДАЦИИ:

Завести счётчик баланса скобок (увеличивается при появлении открывающей скобки, уменьшается при появлении закрывающей скобки). Обратить особое внимание на 2 случая: 1) появление непарной закрывающей скобки, т.е. счётчик стал отрицательным (далее нет смысла продолжать анализ текста, т.е. надо завершать цикл и выдавать ответ); 2) наличие непарных открывающих скобок (отлавливается в конце, по окончании ввода всего текста). Удачный исход: если дошли до точки и при этом счётчик баланса - нулевой.

Для облегчения вывода ответа сделать начальное предположение, что последовательность не сбалансирована (т.е. до начала чтения текста настроиться с помощью, например, регистра **EDX** на начало строки со словом “НЕТ”). Но если в конце выяснится, что баланс скобок имеет место, то перед выводом ответа (по макрокоманде **outstr EDX**) скорректировать значение в **EDX**. Как настраиваться на начало строки с выводимым текстом – написано в рекомендациях к задаче 2.

Задача 4. “Дробь P/Q”

Описать в программе следующие переменные:

P dd ? ; P ≥ 0

Q dd ? ; Q > 0

Ввести с клавиатуры значения для переменных P и Q.

Напечатать дробь P/Q в виде вещественного числа с 5 цифрами в дробной части.

РЕКОМЕНДАЦИИ:

1) Сначала вывести *целую часть* от деления P на Q.

2) Затем вывести *точку* (макрокоманда **outchar ‘.’**).

3) Наконец, запустить цикл (**loop**) из пяти шагов. На каждом шаге будет выводиться очередная цифра дробной части: для её нахождения надо последний найденный остаток домножить на 10, а затем полученный результат разделить на Q. *Внимание:* команда **loop** реализует только короткий переход (то есть тело цикла не может содержать более 30-40 команд). Может так случиться, что придётся отказаться в этой задаче от использования **loop** (заменив её другими командами). Но сначала попробуйте воспользоваться **loop** (надеюсь, повезёт).

4) В этой задаче можно не экономить на числе обращений к оперативной памяти.

Задача 5 . “ Произведение старшей и младшей десятичных цифр ”

Ввести число без знака (по **inint**) и напечатать произведение старшей (значащей) и младшей цифр в десятичной записи этого числа (значащей называется цифра, удаление которой меняет величину числа).

РЕКОМЕНДАЦИИ: Решать путём последовательного деления на 10. Разобраться, какое деление (длинное, короткое или сверхдлинное) здесь необходимо. Не забыть про случай, когда в числе одна цифра (она является одновременно старшей и младшей).

Задача 6 . “ Пятеричное число ”

Ввести (по **inchar**) непустую последовательность цифр, оканчивающуюся пробелом, которая является правильной записью неотрицательного числа в пятеричной системе счисления. Напечатать это число (за одно обращение к команде **outword**) в десятичной системе. Считать, что искомое число уместится в формат двойного слова.

РЕКОМЕНДАЦИИ: Ввод пятеричного числа осуществлять с использованием схемы Горнера. Разобраться, какой вид умножения (короткое, длинное или сверхдлинное) надо использовать.

Задача 7 . “ Ближайшее число, кратное семи ”

Ввести число без знака (по **inint**) и напечатать ближайшее к нему число, кратное семи (возможность выхода из диапазона представимости не учитывать).

РЕКОМЕНДАЦИИ: Разобраться, какое деление (длинное, короткое или сверхдлинное) здесь необходимо. Проверить остаток от деления исходного числа на 7. Если остаток равен 1, 2, 3, то искомое число лежит (на числовой оси) *левее* заданного. Если остаток равен 4, 5, 6, то искомое число лежит (на числовой оси) *правее* заданного. Если остаток нулевой – искомое число совпадает с заданным.

Задача 8 . “ Алгебраическая сумма ”

Дан текст следующего вида:

$$d_1 \pm d_2 \pm \dots \pm d_k .$$

где d_i - цифра от 0 до 9, $k \geq 1$. Найти значение этой алгебраической суммы.

Считывать текст до появления точки. Считать, что результат суммирования укладывается в формат слова. В этой задаче для ввода использовать только макрокоманду **inchar** (т.е. все числа и знаки вводятся *посимвольно*). Значение найденной суммы выводить по макрокоманде **outint**.

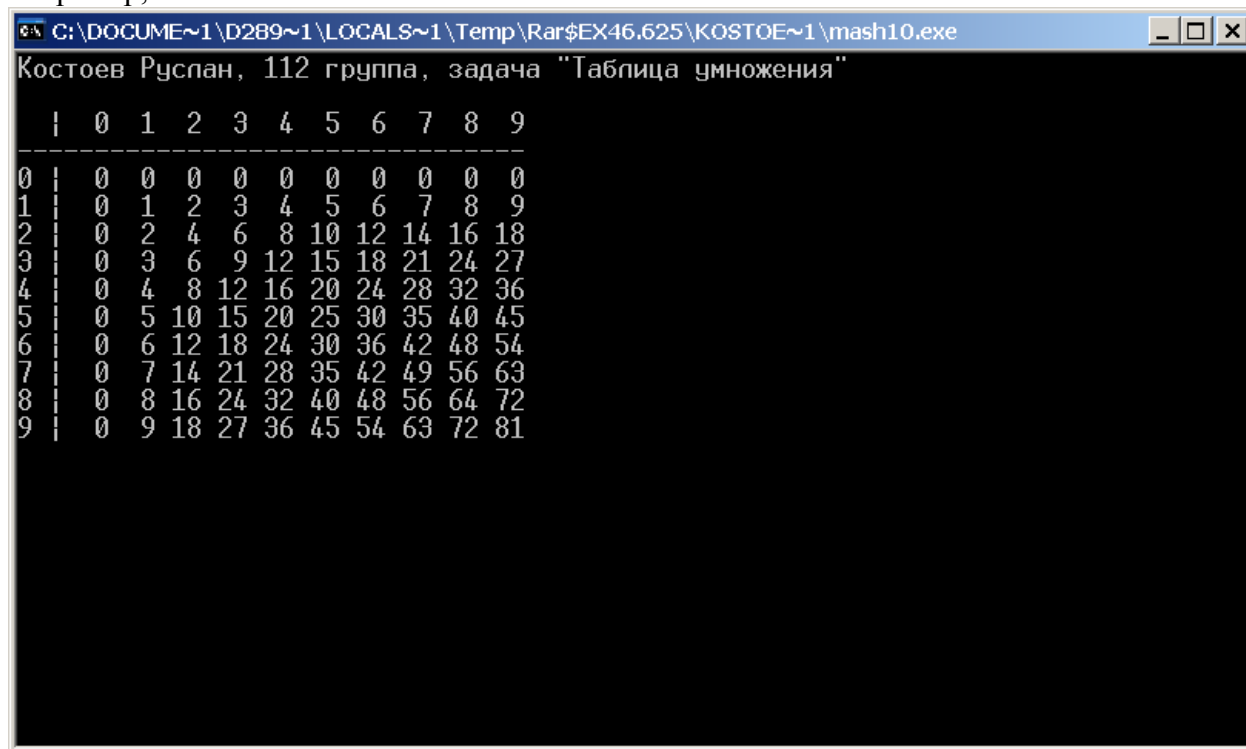
Задача 9 . “Первая и последняя буквы”

Дана непустая последовательность непустых слов из малых латинских букв; между соседними словами - запятая, за последним словом - точка. Подсчитать количество слов, которые начинаются и оканчиваются одной и той же буквой.

РЕКОМЕНДАЦИИ: Реализовать двойной цикл. *Внешний* цикл – по словам; *внутренний* – по символам текущего слова (до точки или запятой). Выйдя из внутреннего цикла – сравнить крайние буквы прочитанного слова. Текст набирать на клавиатуре надо сразу целиком (по аналогии с решением аналогичных задач на Паскале), ставить точку и нажимать **Enter** (предполагается, что текст будет не очень длинным и поэтому целиком “влезет” в буфер ввода). Символы текста вводить с помощью **inchar**.

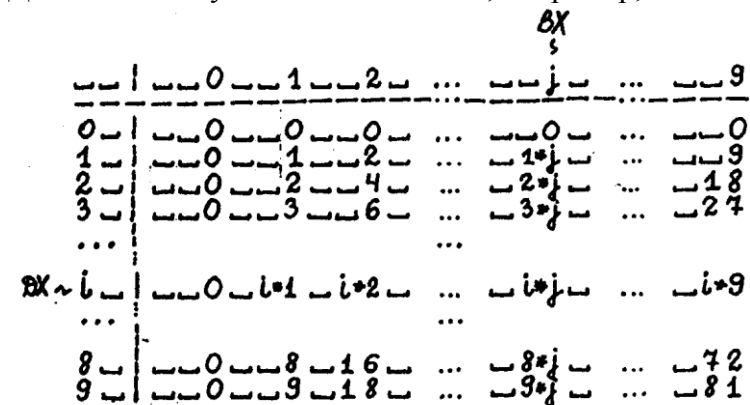
Задача 10. "Таблица умножения"

Напечатать таблицу умножения в десятичной системе счисления. Постараться красиво оформить вывод, например, так:



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|----|----|----|----|----|----|----|----|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 |
| 3 | 0 | 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 27 |
| 4 | 0 | 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 |
| 5 | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 |
| 6 | 0 | 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 |
| 7 | 0 | 7 | 14 | 21 | 28 | 35 | 42 | 49 | 56 | 63 |
| 8 | 0 | 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 | 72 |
| 9 | 0 | 9 | 18 | 27 | 36 | 45 | 54 | 63 | 72 | 81 |

Для этого следует воспользоваться, например, такой схемой:



| | 0 | 1 | 2 | ... | j | ... | 9 |
|-----|---|-----|-----|-----|-----|-----|-----|
| 0 | 0 | 0 | 0 | ... | 0 | ... | 0 |
| 1 | 0 | 1 | 2 | ... | 1*j | ... | 9 |
| 2 | 0 | 2 | 4 | ... | 2*j | ... | 18 |
| 3 | 0 | 3 | 6 | ... | 3*j | ... | 27 |
| ... | | | | ... | | ... | |
| i | 0 | i*1 | i*2 | ... | i*j | ... | i*9 |
| ... | | | | ... | | ... | |
| 8 | 0 | 8 | 16 | ... | 8*j | ... | 72 |
| 9 | 0 | 9 | 18 | ... | 9*j | ... | 81 |

Как видно из схемы, здесь намечается *двойной цикл*: внешний по *i*, внутренний по *j*.

Замечание: первые две строки таблицы выводим вне основного цикла. Для этого, например, можно:

S1 db " | 0 1 2 3 4 5 6 7 8 9 ",0

S2 db 33 dup("-",0)

и в программе распечатать эти строки по **outstrln**.

Всё остальное выводится в цикле единообразно.