МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»**

Кафедра дискретной математики и информатики

ЛАБОРАТОРНАЯ РАБОТА №3

студента 2 курса 221 группы

направления 09.03.01 Информатика и вычислительная техника

факультета компьютерных наук и информационных технологий

Мусатова Федора Алексеевича

Саратов 2025

**Задание 3.1.** В регистре AX задано число от 0 до 65535. Выведите это число на экран. (Проверить программу для числа более 2600.)

.model small ;Модель памяти SMALL использует сегменты

.stack 100h ;Сегмент стека размером 100h (256 байт)

.data ;Начало сегмента данных

Namee db 'Musatov Fedor 221', 0Dh, 0Ah, '$'

;Создание строки для вывода имени и фамилии

.code ;Начало сегмента кода

start: ;Точка входа в программу

mov AX, @data ;Помещаем указатель на сегмент данных в AX

mov DS, AX ;Помещаем указатель на сегмент данных в DS

mov AH, 09h ;Код команды прерывания для вывода строки помещаем в AH

mov DX, offset Namee ;В регистр DX помещаем адрес начала строки

int 21h ;Вызов команды прерывания

mov AX, 3000 ;Загружает в регистр AX число 3000 (будет преобразовано в десятичное)

mov BX, 10 ;Загружает в регистр BX значение 10 (основание системы счисления)

mov CX, 0 ;Обнуляет регистр CX, который будет использоваться как счетчик

loopf:

inc CX ;Увеличивает значение в регистре CX на 1

mov DX, 0 ;Обнуляет регистр DX, который нужен для стека

div BX ;Делит значение в регистре AX на значение в регистре BX

push DX ;Записывает остаток (DX) в стек

cmp AX, 0 ;Сравнивает значение в AX с 0

jne loopf ;Если значение в AX не равно 0 (переход на метку loopf), то цикл продолжается

loops:

mov AH, 02h ;Загружает в регистр AH код функции DOS для вывода символа

pop DX ;Извлекает значение из стека и помещает его в регистр DX

call print ;Вызов процедуры

loop loops ;Продолжает цикл, пока не завершится

mov AX, 4C00h ;Функция 4Ch завершения программы с кодом возврата 0

int 21h ;Вызов функции DOS

print proc

add DX, 30h ;Преобразуем цифру в ASCII-символ

int 21h ;Вызов прерывания

ret ;Возврат в процедуру

print endp ;Конец процедуры

end start ;Конец текста программы с точкой входа

**Задание 2**

Первая цифра задана в AX, вторая цифра задана в BX. Написать программу, которая выводит в одну строку первую цифру (AX), пробел, вторую цифру (BX). Далее совершает обмен значений регистров AX и BX и снова в новой строке на экране выводит в одну строку первую цифру (AX), пробел, вторую цифру (BX). Обмен совершить без использования дополнительной памяти (командой XCHG). **Структура программы должна обязательно содержать одну или более вспомогательных процедур.**

.model small ;Модель памяти SMALL использует сегменты

.stack 100h ;Сегмент стека размером 100h (256 байт)

.data ;Начало сегмента данных

Namee db 'Musatov Fedor 221', 0Dh, 0Ah, '$'

;Создание строки для вывода имени и фамилии

.386

.code ;Начало сегмента кода

start: ;Точка входа в программу

mov AX, @data ;Помещаем указатель на сегмент данных в AX

mov DS, AX ;Помещаем указатель на сегмент данных в DS

mov AH, 09h ;Код команды прерывания для вывода строки помещаем в AH

mov DX, offset Namee ;В регистр DX помещаем адрес начала строки

int 21h ;Вызов команды прерывания

mov eax, 65536 ;Загружает в регистр EAX число 65536 (будет преобразовано в десятичное)

mov ebx, 10 ;Загружает в регистр EBX значение 10 (основание системы счисления)

mov CX, 0 ;Обнуляет регистр CX, который будет использоваться как счетчик

loopf:

inc CX ;Увеличивает значение в регистре CX на 1

mov edx, 0 ;Обнуляет регистр EDX перед делением

div ebx ;Делит значение в регистре EAX на значение в регистре EBX

push edx ;Записывает остаток (EDX) в стек

cmp eax, 0 ;Сравнивает значение в EAX с 0

jne loopf ;Если значение в EAX не равно 0 (переход на метку loopf), то цикл продолжается

loops:

mov AH, 02h ;Загружает в регистр AH код функции DOS для вывода символа

pop edx ;Извлекает значение из стека и помещает его в регистр EDX

call print ;Вызов процедуры

loop loops ;Продолжаем цикл, пока не завершится

mov eax, 4C00h ;Функция 4Ch завершения программы с кодом возврата 0

int 21h ;Вызов функции DOS

print proc

add edx, 30h ;Преобразуем цифру в ASCII-символ

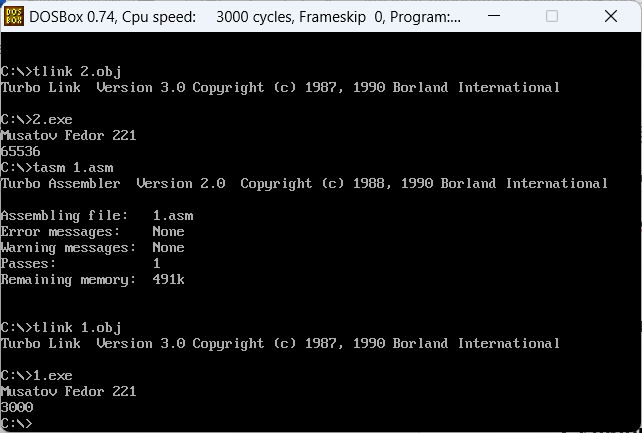
int 21h ;Вызов прерывания

ret ;Возврат в процедуру

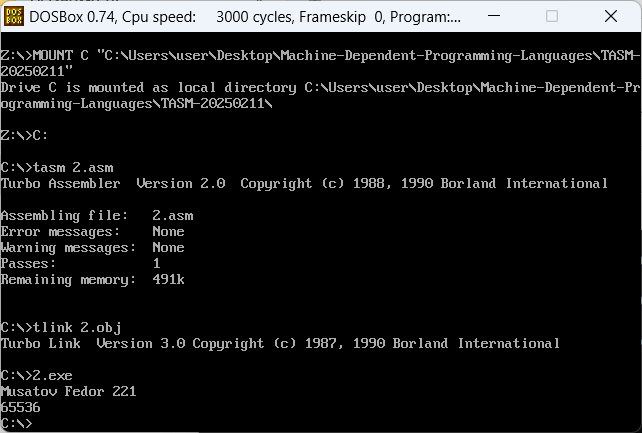
print endp ;Конец процедуры

end start ;Конец текста программы с точкой входа

**Скриншот с запуска 1 программы**



**Скриншот с запуска 2 программы**



**Ответы на контрольные вопросы**

1. Чем отличается деление на байт от деления на слово? (где должно располагаться делимое, куда попадут частное от деления и остаток от деления)

Байтовая команда делит 16-битовое делимое на 8-битовый делитель. Делимое находится в регистре AX. В результате деления получается два числа: частное помещается в регистр AL, а остаток – в AH.

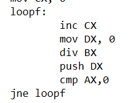
Команда, работающая со словами, делит 32-битовое делимое на 16-битовый делитель. Делимое находится в паре регистров DX: AX, причем регистр DX содержит старшую значимую часть, а регистр AX – младшую. Команда деления помещает частное в регистр AX, а остаток в DX.



Ни один из флагов состояния не определен после команды деления. Однако, если частное больше того, что может быть помещено в регистр результата (255 для байтового деления и 65535 для деления слов), возникает ошибка значимости и выполняется программное прерывание уровня 0.

1. Каков механизм действия команды cmp? В паре с какими командами она обычно используется?

Команда сравнения CMP сравнивает два числа, вычитая второе из первого. Инструкция CMP не сохраняет результат, а лишь устанавливает в соответствии с результатом флаги состояния. Основное назначение команды CMP – это организация ветвлений (условных переходов) в ассемблерных программах. Эта команда часто используется с командами условного перехода.



1. На какие флаги реагируют команды условного перехода для чисел со знаком и для чисел без знака?

Первая группа команд Jcc (кроме JCXZ/JECXZ) проверяет текущее состояние регистра флагов (не изменяя его) и в случае соблюдения условия осуществляет переход на смещение, указанное в качестве операнда. Флаги, проверяемые командой, кодируются в ее мнемонике, например: JC – переход, если установлен CF. Сокращения «L» ( less – меньше) и «G» ( greater – больше) применяются для целых со знаком, а «A» ( above – над) и «B» ( below – под) для целых без знака. Ниже в таблице показаны команды условного перехода и проверяемые ими флаги.



Буква Х в любой позиции означает, что команда не проверяет флаг. Цифра 0 означает, что флаг должен быть сброшен, а цифра 1 означает, что флаг должен быть установлен, чтобы условие было выполнено (переход произошел).

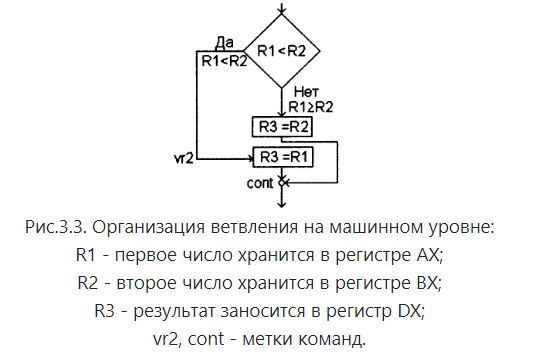
Команды условного перехода можно разделить на три подгруппы:

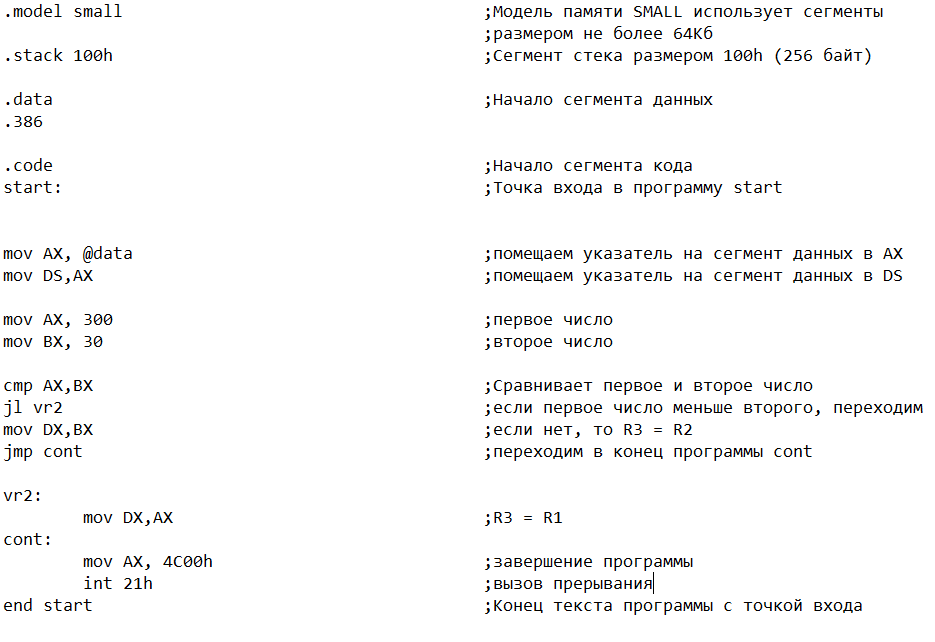
1) Непосредственно проверяющие один из флагов на равенство 0 или 1.

2) Арифметические сравнения со знаком. Существуют 4 условия, которые могут быть проверены: меньше (JL), меньше или равно (JLE), больше (JG), больше или равно (JGE). Эти команды проверяют одновременно три флага: знака, переполнения и нуля.

3) Арифметические без знака. Здесь также существует 4 возможных соотношения между операндами: меньше (JB), меньше или равно (JBE), больше (JA), больше или равно (JAE). Учитываются только два флага. Флаг переноса показывает какое из двух чисел больше. Флаг нуля определяет равенство.

1. С помощью команд условного и безусловного перехода выполните программную реализацию алгоритма ветвления для определения наименьшего числа из двух заданных. Алгоритм изображен в виде блок-схемы, приведенной на рис.3.3.





1. Каков механизм работы команды организации цикла LOOP?

Все команды цикла используют регистр CX в качестве счетчика цикла. Простейшая из них – команда LOOP. Она в конце каждой итерации уменьшает содержимое CX на 1 и передает управление на метку (указанную в команде), если содержимое CX не равно 0. Если вычитание 1 из CX привело к нулевому результату, выполняется следующая команда.

Команда LOOPNE (цикл пока не равно) выходит из цикла, если не установлен флаг нуля или если в регистре CX получился 0. Команда LOOPE (цикл пока равно) выполняет обратную к описанной проверку флага нуля: цикл здесь завершается, если регистр CX достиг 0 или если установлен флаг 0.

1. Как с помощью команды сдвига можно умножить знаковое число, хранящееся в АХ, на 2 в n-ой степени?

С помощью команды сдвига влево SAL (или SHL). В результате работы этой команды при сдвиге влево на n позиций число умножается на 2^n.

Пример:

SAL AX,CL (CL = n)

MOV CX, 2 ;n = 2 (если сдвигаем на две позиции, умножаем на 2^2 = 4)

SAL AX, CL ;Арифметический сдвиг AX на CL (CX) позиций

1. Как с помощью команды сдвига проверить содержимое регистра ВХ на четность?

С помощью команд сдвига (SAR, SHR, ROR, RCR) можно проверить содержимое BX на четность таким образом: сдвиг переносит младший бит числа в флаг CF (он отражает состояние младшего бита после сдвига вправо).

Младший бит в двоичном представлении числа определяет его четность. Если младший бит 1, то число нечетное. Если младший бит 0, то число четное.

SHR BX, 1 ; Сдвигаем BX вправо на одну позицию

Пример: Число 111101 в двоичной системе счисления. Младший бит этого числа - 1. Сдвиг вправо на 1 позицию: 111101 >> 1 = 011110

В результате сдвига, младший бит 1 переходит в флаг (CF). CF становится установленным (CF = 1).

Так как флаг установился в (CF = 1), мы делаем вывод, что исходное число 111101 было нечетным. Аналогично для четных чисел.