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 №2

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 1**

Первая цифра задана в AX, вторая цифра задана в BX. Написать программу, которая выводит в одну строку первую цифру, пробел, вторую цифру.

**Текст 1 программы**

.model tiny ;Модель памяти TINY, в которой код, данные и стек

;размещаются в одном и том же сегменте размером до 64Кб

.code ;Начало сегмента кода

org 100h ;Устанавливает значение программного счетчика в 100h

;Начало необходимое для COM-программы,

;которая загружается в память с адреса PSP:100h

start:

mov DX, offset Hello;помещаем адрес строки Hello для вывода в консоль

mov AH,09h ;перемещаем код прерываемой команды, чтобы вывести строку

int 21h ;вызов директивы прерывания

mov AX, 01h ;первая цифра

mov BX, 04h ;вторая цифра

mov DL, '0' ;ввод нужной разницы для получения ASCII кода в режиме вывода

add DL, AL ;получение кода ASCII первой цифры в режиме вывода

mov AH, 02h ;помещаем код прерываемой программы для вывода символа

int 21h ;вызов директивы прерывания

mov DX, offset probel;помещаем адрес строки probel для вывода в консоль

mov AH, 09h ;перемещаем код прерываемой команды, чтобы вывести строку

int 21h ;вызов директивы прерывания

mov DL, '0' ;ввод нужной разницы для получения ASCII кода в режиме вывода

add DL, BL ;получение кода ASCII второй цифры в режиме вывода

mov AH, 02h ;помещаем код прерываемой программы для вывода символа

int 21h ;вызов директивы прерывания

mov AX, 4C00h;завершение программы

int 21h ;вызов директивы прерывания

;===== Data =====

Hello db 'Musatov Fedor 221', 0Dh, 0Ah, '$' ;инициализация строки hello

probel db ' $' ;инициализация строки probel

end start ;конец программы

**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 tiny ;Модель памяти TINY, в которой код, данные и стек

;размещаются в одном и том же сегменте размером до 64кб

.code ;Начало сегмента кода

org 100h ;Устанавливает значение программного счетчика в 100h

;Начало необходимое для COM-программы,

;которая загружается в память с адреса PSP:100h

start:

mov DX,offset Hello;помещаем адрес строки Hello для вывода в консоль

call OutString ;вызов процедуры

mov AX, 01h ;первая цифра

mov BX, 04h ;вторая цифра

mov DL, '0' ;ввод нужной разницы для получения ASCII кода в режиме вывода

add DL, AL ;получение кода ASCII первой цифры в режиме вывода

call OutChar ;вызов процедуры

mov DX, offset probel ;помещаем адрес строки probel для вывода в консоль

call Outstring ;вызов процедуры

mov DL, '0' ;ввод нужной разницы для получения ASCII кода в режиме вывода

add DL, BL ;получение кода ASCII второй цифры в режиме вывода

call OutChar ;вызов процедуры

mov DX, offset newstr ;помещаем адрес строки probel для вывода в консоль

call Outstring ;вызов процедуры

XCHG AX,BX ;меняем местами первую и вторую цифру

mov DL, '0' ;ввод нужной разницы для получения ASCII кода в режиме вывода

add DL, AL ;получение кода ASCII второй цифры в режиме вывода

call OutChar ;вызов процедуры

mov DX,offset probel;помещаем адрес строки probel для вывода в консоль

Call Outstring ;вызов процедуры

mov DL, '0' ;ввод нужной разницы для получения ASCII кода в режиме вывода

add DL, BL ;получение кода ASCII второй цифры в режиме вывода

call OutChar ;вызов процедуры

mov AX, 4C00h ; завершение программы

int 21h ;вызов директивы прерывания

OutString proc ;процедура вывода строки

mov AH, 09h; помещаем код прерываемой программы для вывода строки

int 21h ;вызов директивы прерывания

ret ;конец процедуры

OutString endp

OutChar proc ;процедура вывода символа

push AX ; Сохраняем AX в стеке

push BX ; Сохраняем BX в стеке

mov AH, 02h; помещаем код прерываемой программы для вывода символа

int 21h ;вызов директивы прерывания

pop BX

pop AX

ret ;конец процедуры

OutChar endp

;===== Data =====

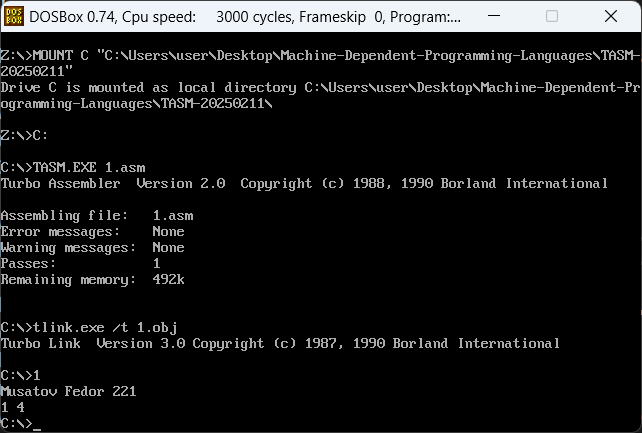
Hello db 'Musatov Fedor 221', 0Dh, 0Ah, '$' ;инициализация строки hello

probel db ' $' ;инициализация строки probel

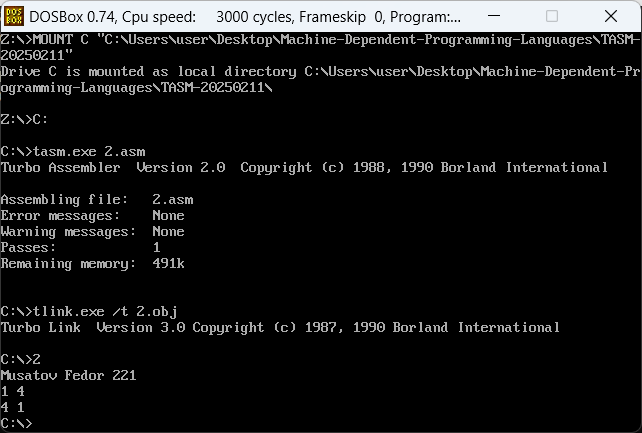
newstr db '',0Dh, 0Ah, '$' ;инициализация строки newstr

end start ;конец программы

**Скриншот с запуска 1 программы**



**Скриншот с запуска 2 программы**



**Таблица 1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Шаг | Машинный код | Команды | Регистры | | | | | | | | | Флаги |
|  |  |  | AX | BX | CX | DX | SP | DS | SS | CS | IP | CZSOPAID |
| 0 |  |  | 0000 | 0000 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0100 | 00000010 |
| 1 | BA2901 | Mov dx,0129 | 0000 | 0000 | 0000 | 0129 | FFFE | 489D | 489D | 489D | 0103 | 00000010 |
| 2 | B409 | Mov ah,09 | 0900 | 0000 | 0000 | 0129 | FFFE | 489D | 489D | 489D | 0105 | 00000010 |
| 3 | CD21 | Int 21 | 0900 | 0000 | 0000 | 0129 | FFFE | 489D | 489D | 489D | 0107 | 00000010 |
| 4 | B80800 | Mov ax,0001 | 0001 | 0000 | 0000 | 0129 | FFFE | 489D | 489D | 489D | 010A | 00000010 |
| 5 | BB0900 | Mov bx,0004 | 0001 | 0004 | 0000 | 0129 | FFFE | 489D | 489D | 489D | 010D | 00000010 |
| 6 | B230 | Mov dl,30 | 0001 | 0004 | 0000 | 0130 | FFFE | 489D | 489D | 489D | 010F | 00000010 |
| 7 | 02D0 | Add dl,al | 0001 | 0004 | 0000 | 0131 | FFFE | 489D | 489D | 489D | 0111 | 00000010 |
| 8 | B402 | Mov ah,02 | 0201 | 0004 | 0000 | 0131 | FFFE | 489D | 489D | 489D | 0113 | 00000010 |
| 9 | CD21 | Int 21 | 0231 | 0004 | 0000 | 0138 | FFFE | 489D | 489D | 489D | 0115 | 00000010 |
| 10 | BA4001 | Mov dx,0140 | 0231 | 0004 | 0000 | 013D | FFFE | 489D | 489D | 489D | 0118 | 00000010 |
| 11 | B409 | Mov ah,09 | 0931 | 0004 | 0000 | 013D | FFFE | 489D | 489D | 489D | 011A | 00000010 |
| 12 | CD21 | Int 21 | 0931 | 0004 | 0000 | 013D | FFFE | 489D | 489D | 489D | 011C | 00000010 |
| 13 | B230 | Mov dl,30 | 0931 | 0004 | 0000 | 0130 | FFFE | 489D | 489D | 489D | 011E | 00000010 |
| 14 | 02D3 | Add dl,bl | 0931 | 0004 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0120 | 00001010 |
| 15 | B402 | Mov ah,02 | 0231 | 0004 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0122 | 00001010 |
| 16 | CD21 | Int 21 | 0234 | 0004 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0124 | 00001010 |
| 17 | B8004C | Mov ax,4C00 | 4C00 | 0004 | 0000 | 0139 | FFFE | 489D | 489D | 489D | 0127 | 00001010 |
| 18 | CD21 | Int 21 | 0192 | 2110 | F668 | 2110 | 0106 | 2110 | 0192 | 0000 | 0000 | 10100011 |

**Таблица 2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Шаг | Машинный код | Команды | Регистры | | | | | | | | | Флаги |
|  |  |  | AX | BX | CX | DX | SP | DS | SS | CS | IP | CZSOPAID |
| 0 |  |  | 0000 | 0000 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0100 | 00000010 |
| 1 | BA5001 | Mov dx,0150 | 0000 | 0000 | 0000 | 0150 | FFFE | 489D | 489D | 489D | 0103 | 00000010 |
| 2 | E84000 | Call 0146 | 0900 | 0000 | 0000 | 0150 | FFFЕ | 489D | 489D | 489D | 0106 | 00000010 |
| 3 | B80100 | Mov  ax,0001 | 0001 | 0000 | 0000 | 0150 | FFFE | 489D | 489D | 489D | 0109 | 00000010 |
| 4 | BB0400 | Mov  bx,0004 | 0001 | 0004 | 0000 | 0150 | FFFE | 489D | 489D | 489D | 010C | 00000010 |
| 5 | B230 | Mov dl,30 | 0001 | 0004 | 0000 | 0130 | FFFE | 489D | 489D | 489D | 010E | 00000010 |
| 6 | 02D0 | Add dl, al | 0001 | 0004 | 0000 | 0131 | FFFE | 489D | 489D | 489D | 0110 | 00000010 |
| 7 | E83800 | Call 014B | 0231 | 0004 | 0000 | 0131 | FFFE | 489D | 489D | 489D | 0113 | 00000010 |
| 8 | BA6401 | Mov dx,0164 | 0231 | 0004 | 0000 | 0164 | FFFE | 489D | 489D | 489D | 0116 | 00000010 |
| 9 | T82D00 | Call 0146 | 0931 | 0004 | 0000 | 0164 | FFFE | 489D | 489D | 489D | 0119 | 00000010 |
| 10 | B230 | Mov dl,30 | 0931 | 0004 | 0000 | 0130 | FFFE | 489D | 489D | 489D | 011B | 00000010 |
| 11 | 02D3 | Add dl,bl | 0931 | 0004 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 011D | 00000010 |
| 12 | E82B00 | Call 014B | 0234 | 0004 | 0000 | 0134 | FFFЕ | 489D | 489D | 489D | 0120 | 00000010 |
| 13 | BA6601 | Mov dx,0166 | 0234 | 0004 | 0000 | 0166 | FFFЕ | 489D | 489D | 489D | 0123 | 00000010 |
| 14 | E82000 | Call 0146 | 0934 | 0004 | 0000 | 0166 | FFFE | 489D | 489D | 489D | 0126 | 00000010 |
| 15 | B80100 | Mov ax,0001 | 0001 | 0004 | 0000 | 0166 | FFFE | 489D | 489D | 489D | 0129 | 00000010 |
| 16 | BB0400 | Mov bx,0004 | 0001 | 0004 | 0000 | 0166 | FFFE | 489D | 489D | 489D | 012C | 00000010 |
| 17 | 93 | Xchg bx,ax | 0004 | 0001 | 0000 | 0166 | FFFE | 489D | 489D | 489D | 012D | 00000010 |
| 18 | B230 | Mov dl, 30 | 0004 | 0001 | 0000 | 0130 | FFFE | 489D | 489D | 489D | 012F | 00000010 |
| 19 | 02D0 | Add dl,al | 0234 | 0001 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0131 | 00000010 |
| 20 | EB1700 | Call 014B | 0234 | 0001 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0134 | 00000010 |
| 21 | BA6401 | Mov dx,0164 | 0234 | 0001 | 0000 | 0164 | FFFE | 489D | 489D | 489D | 0137 | 00000010 |
| 22 | EB0C00 | Call 0146 | 0934 | 0001 | 0000 | 0164 | FFFE | 489D | 489D | 489D | 013A | 00000010 |
| 23 | B230 | Mov dl,30 | 0934 | 0001 | 0000 | 0130 | FFFE | 489D | 489D | 489D | 013C | 00000010 |
| 24 | 02D3 | Add dl,bl | 0231 | 0001 | 0000 | 0131 | FFFE | 489D | 489D | 489D | 013E | 00000010 |
| 25 | EB0A00 | Call 014B | 4C00 | 0001 | 0000 | 0131 | FFFE | 489D | 489D | 489D | 0141 | 00000010 |
|  | B8004C | Mov ax,4C00 | 0192 | F644 | F62E | 0206 | 0106 | 2110 | 0192 | 0000 | 0144 | 00000010 |

**Ответы на контрольные вопросы**

1. **В какой регистр надо поместить код выводимого символа? Какой код Dos-функции используется для вывода отдельного символа на экран?**

Код выводимого символа нужно поместить в DL. Для вывода отдельного символа используется код 02h.

Например:

add DL, AL

mov AH, 02h

1. **Какая операция позволяет получить для цифры её код в кодовой таблице?**

Для вывода на экран цифры необходимо сначала преобразовать ее в символьную форму. То есть для получения символьной формы необходимо заменить цифру кодом ASCII ее изображения. Операция add позволяет получить для цифры ее код в кодовой таблице. В регистр DL записывается код символа 0, и с помощью команды ADD к содержимому DL прибавляется смещение для получения нужной цифры. Например, mov DL,'0'

add DL, AL

1. **Объясните назначение процедуры. Как определяются начало и конец процедуры?**

Все современные программы разрабатываются по модульному принципу – программа обычно состоит из одной или нескольких небольших частей, называемых подпрограммами или процедурами, и одной главной программы, которая вызывает эти процедуры на выполнение, передавая им управление процессором. После завершения работы процедуры возвращают управление главной программе и выполнение продолжается с команды, следующей за командой вызова подпрограммы.

Достоинством такого метода является возможность разработки программ значительно большего объема небольшими функционально законченными частями. Кроме того, эти подпрограммы можно использовать в других программах, не прибегая к переписыванию частей программного кода. В довершение ко всему, так как размер сегмента не может превышать 64К, то при разработке программ с объемом кода более 64К, просто не обойтись без модульного принципа.

Язык программирования ассемблера поддерживает применение процедур двух типов – ближнего (near) и дальнего (far).

Процедуры ближнего типа должны находиться в том же сегменте, что и вызывающая программа.

Дальний тип процедуры означает, что к ней можно обращаться из любого другого кодового сегмента.

При вызове процедуры в стеке сохраняется адрес возврата в вызывающую программу:

- при вызове ближней процедуры – слово, содержащее смещение точки возврата относительно текущего кодового сегмента;

- при вызове дальней процедуры – слово, содержащее адрес сегмента, в котором расположена точка возврата, и слово, содержащее смещение точки возврата в этом сегменте.

Описание процедуры имеет следующий синтаксис:

<имя\_процедуры>PROC<параметр> Начало

<тело\_процедуры>

RET ;Возврат из подпрограммы в точку вызова

<имя\_процедуры> ENDP Конец

Параметр, указываемый после ключевого слова PROC, определяет тип процедуры: ближний (NEAR) или дальний (FAR). Если параметр отсутствует, то по умолчанию процедура считается ближней.

1. **Ваша программа состоит из главной процедуры и процедур-подпрограмм. Каким может быть взаимное расположение главной процедуры и подпрограмм?**

В общем случае, размещать подпрограмму в теле программы можно где угодно, но при этом следует помнить, что сама по себе подпрограмма выполняться не должна, а должна выполняться лишь при обращении к ней. Поэтому подпрограммы принято размещать либо в конце сегмента кода, после команд завершения программы, либо в самом начале сегмента кода, перед точкой входа в программу. В больших программах подпрограммы нередко размещают в отдельном кодовом сегменте.

1. **Как процессор использует стек при работе с любой процедурой?**

Все команды вызова CALL безусловны. Внутрисегментный вызов NEAR CALL используется для передачи управления процедуре, находящейся в том же сегменте. Он указывает новое значение регистра IP и сохраняет в стеке адрес возврата, т.е. IP команды, следующей за командой CALL.

Межсегментный вызов FAR CALL используется для передачи управления процедуре, находящейся в другом сегменте или даже программном модуле. Он задает новые значения сегмента CS и смещения IP для дальнейшего выполнения программы и сохраняет в стеке как регистр IP, так и регистр CS.

Все возвраты RET являются косвенными переходами, поскольку извлекают адрес перехода из вершины стека. Внутрисегментный возврат извлекает из стека одно слово и помещает его в регистр IP, а межсегментный возврат извлекает из стека два слова, помещая слова из меньшего адреса в регистр IP, а слово из большего адреса – в регистр CS.

1. **С помощью какой команды вызывается процедура? Как меняется значение регистра SP после вызова процедуры? Приведите пример из вашей таблицы трассировки.**

Процедура вызывается с помощью команды Call. При ее вызове значение SP уменьшается. (в стек помещаются значения регистра IP, если NEAR; IP и CS, если FAR)

1. **После какой команды процедуры из стека извлекается адрес возврата?**

Инструкция RET, относящаяся к процедуре типа FAR, будет транслироваться ассемблером как команда RETF, которая извлекает из стека значения для регистров IP и CS.