

Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

**МАШИННО-ЗАВИСИМЫЕ ЯЗЫКИ ПРОГРАММИРОВАНИЯ.
ЛАБОРАТОРНАЯ РАБОТА №2**

ОТЧЁТ

студентки 2 курса 251 группы
направления 09.03.04 — Программная инженерия
факультета компьютерных наук и информационных технологий
Потапкиной Маргариты Андреевны

Проверено:

старший преподаватель

Е. М. Черноусова

СОДЕРЖАНИЕ

1	Текст задания	3
2	Тексты программ на языке ассемблера с комментариями	4
2.1	Программа 1	4
2.2	Программа 2	4
3	Скриншоты запуска программ	5
4	Таблицы трассировки программ с заходом внутрь вызываемых процедур (достаточно одного захода)	6
5	Ответы на контрольные вопросы	7

1 Текст задания

Задание 2.1. Первая цифра задана в AX, вторая цифра задана в BX. Написать программу, которая выводит в одну строку первую цифру, пробел, вторую цифру.

Задание 2.2. 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, регистров. Структура программы должна обязательно содержать одну или более вспомогательных процедур.

Замечание. Команда обмена данных XCHG меняет местами содержимое двух операндов. Порядок следования операндов не имеет значения. В качестве операндов могут выступать регистры (кроме сегментных) и ячейки памяти.

2 Тексты программ на языке ассемблера с комментариями

2.1 Программа 1

```
.model tiny ; модель памяти tiny
.code
org 100h
start:
; вывод строки на экран
mov AH,09h
mov DX,offset Info
int 21h
; вывод цифры из регистра AX на экран
mov AX,8h
add AL,30h
mov DL,AL
mov AH,02h
int 21h
; вывод пробела на экран
mov AX,0h
mov DL,AL
mov AH,02h
int 21h
; вывод цифры из регистра BX на экран
mov BX,3h
add BL,30h
mov DL,BL
mov AH,02h
int 21h
; завершение программы
mov AX,4C00h
int 21h
; Строка с именем, фамилией и переводом строки
Info db 'Potapkina Margarita, 251',0Dh,0Ah,'$'
end start
```

2.2 Программа 2

```
; сегмент стека
stack segment stack 'stack'
db 256 dup (?)
stack ends
; сегмент данных
data segment 'data'
Info db 'Potapkina Margarita, 251$' ; строка с именем и фамилией
```

```

NewLine db ' ',0Dh,0Ah,'$' ; перевод строки
data ends
; сегмент кода
code segment 'code'
assume CS:code,DS:data,SS:stak
; процедура для вывода цифры на экран
PrintDigit proc
    add DL,30h
    push AX
    mov AH,02h
    int 21h
    pop AX
    ret
PrintDigit endp
; процедура для вывода пробела на экран
PrintSpace proc
    push AX
    mov AX,0h
    mov DL,AL
    mov AH,02h
    int 21h
    pop AX
    ret
PrintSpace endp
; процедура для вывода строки на экран
PrintString proc
    push AX
    mov AH,09h
    int 21h
    pop AX
    ret
PrintString endp
start:
; обязательная инициализация регистра DS
mov AX,data
mov DS,AX
; вывод строки с именем и номером группы
mov DX,offset Info
call PrintString
; вывод перевода строки
mov DX,offset NewLine
call PrintString

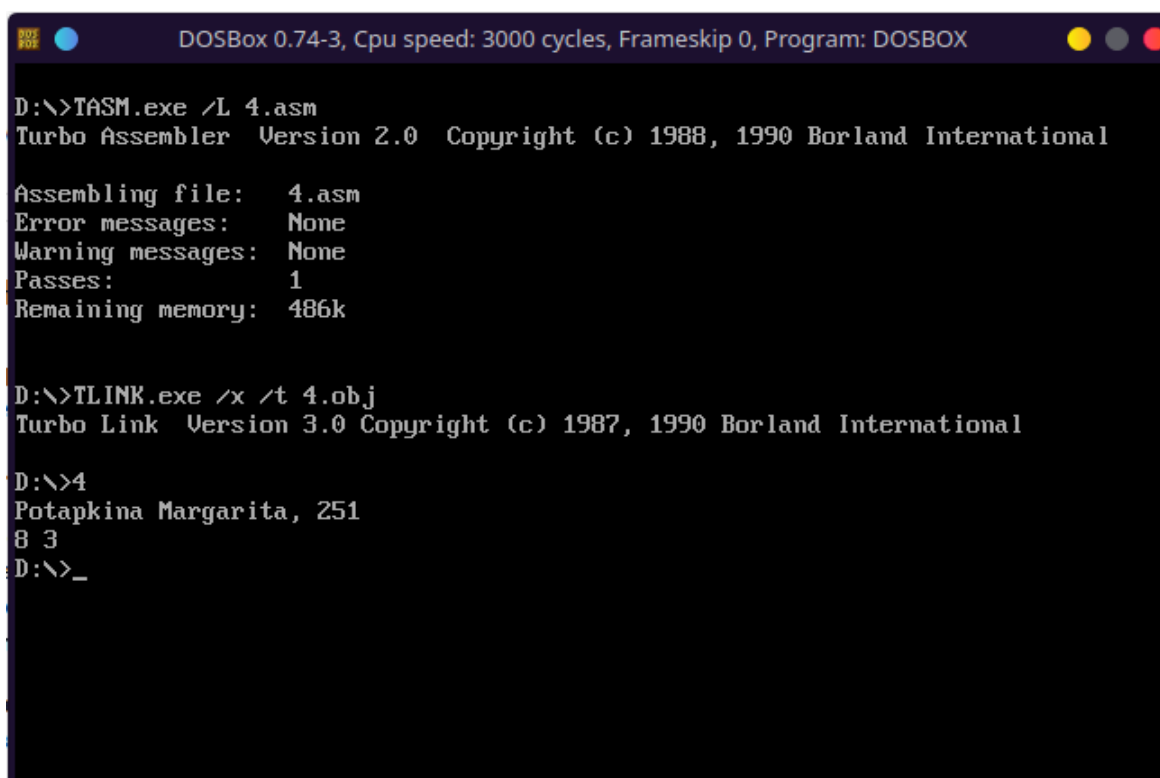
```

```

; вывод цифры из регистра AX
mov AX,07h
mov DL,AL
call PrintDigit
; вывод пробела
call PrintSpace
; вывод цифры из регистра BX
mov BX,05h
mov DL,BL
call PrintDigit
; вывод перевода строки
mov DX,offset NewLine
call PrintString
; обмен значений регистров AX и BX и выполнение тех же действий
XCHG AX,BX
mov DL,AL
call PrintDigit
call PrintSpace
mov DL,BL
call PrintDigit
;завершение программы
mov AX,4C00h
int 21h
code ends
end start

```

3 Скриншоты запуска программ



```
DOSBOX 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

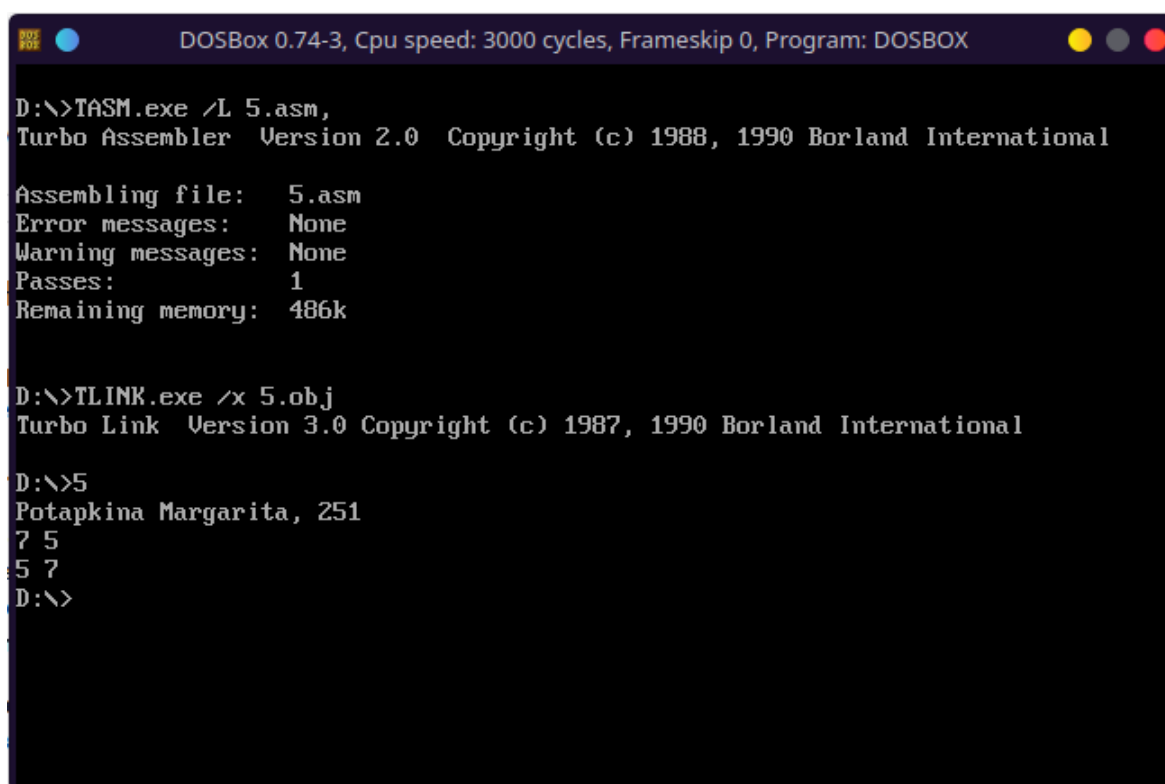
D:\>TASM.exe /L 4.asm
Turbo Assembler Version 2.0 Copyright (c) 1988, 1990 Borland International

Assembling file: 4.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 486k

D:\>TLINK.exe /x /t 4.obj
Turbo Link Version 3.0 Copyright (c) 1987, 1990 Borland International

D:\>4
Potapkina Margarita, 251
8 3
D:\>_
```

Рисунок 3.1 – Скриншот запуска программы 1



```
DOSBOX 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

D:\>TASM.exe /L 5.asm,
Turbo Assembler Version 2.0 Copyright (c) 1988, 1990 Borland International

Assembling file: 5.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 486k

D:\>TLINK.exe /x 5.obj
Turbo Link Version 3.0 Copyright (c) 1987, 1990 Borland International

D:\>5
Potapkina Margarita, 251
7 5
5 7
D:\>
```

Рисунок 3.2 – Скриншот запуска программы 2

4 Таблицы трассировки программ с заходом внутрь вызываемых процедур (достаточно одного захода)

Шаг	Машинный код	Команда	Регистры									Флаги CZSOPAIID
			AX	BX	CX	DX	SP	DS	SS	CS	IP	
1	B409	MOV AH,09	0900	0000	0000	0000	FFFE	489D	489D	489D	0102	00000010
2	BA2C01	MOV DX,012C	0900	0000	0000	012C	FFFE	489D	489D	489D	0105	00000010
3	CD21	INT 21	0900	0000	0000	012C	FFFE	489D	489D	489D	0107	00000010
4	B80800	MOV AX,0008	0008	0000	0000	012C	FFFE	489D	489D	489D	010A	00000010
5	0430	ADD AL,30	0038	0000	0000	012C	FFFE	489D	489D	489D	010C	00000010
6	8AD0	MOV DL,AL	0038	0000	0000	0138	FFFE	489D	489D	489D	010E	00000010
7	B402	MOV AH,02	0238	0000	0000	0138	FFFE	489D	489D	489D	0110	00000010
8	CD21	INT 21	0238	0000	0000	0138	FFFE	489D	489D	489D	0112	00000010
9	B80000	MOV AX,0000	0000	0000	0000	0138	FFFE	489D	489D	489D	0115	00000010
10	8AD0	MOV DL,AL	0000	0000	0000	0100	FFFE	489D	489D	489D	0117	00000010
11	B402	MOV AH,02	0200	0000	0000	0100	FFFE	489D	489D	489D	0119	00000010
12	CD21	INT 21	0200	0000	0000	0100	FFFE	489D	489D	489D	011B	00000010
13	BB0300	MOV BX,0003	0200	0003	0000	0100	FFFE	489D	489D	489D	011E	00000010
14	80C330	ADD BL,30	0200	0033	0000	0100	FFFE	489D	489D	489D	0121	00001010
15	8AD3	MOV DL,BL	0200	0033	0000	0133	FFFE	489D	489D	489D	0123	00001010
16	B402	MOV AH,02	0200	0033	0000	0133	FFFE	489D	489D	489D	0125	00001010
17	CD21	INT 21	0233	0033	0000	0133	FFFE	489D	489D	489D	0127	00001010
18	B8004C	MOV AX,4C00	4C00	0033	0000	0133	FFFE	489D	489D	489D	012A	00001010
19	CD21	INT 21	0192	2110	F674	0BB2	0106	2110	0192	0000	0000	10100011
Шаг	Машинный код	Команда	Регистры									Флаги CZSOPAIID
			AX	BX	CX	DX	SP	DS	SS	CS	IP	
1	B8BD48	MOV AX,48BD	48BD	0000	0000	0000	0100	489D	48AD	48BF	0020	00000010
2	8ED8	MOV DS,AX	48BD	0000	0000	0000	0100	48BD	48AD	48BF	0022	00000010
3	BA0000	MOV DX,0000	48BD	0000	0000	0000	0100	48BD	48AD	48BF	0025	00000010
4	E8EEFF	CALL 0016	48BD	0000	0000	0000	00FE	48BD	48AD	48BF	0016	00000010
5	50	PUSH AX	48BD	0000	0000	0000	00FC	48BD	48AD	48BF	0017	00000010
6	B409	MOV AH,09	09BD	0000	0000	0000	00FC	48BD	48AD	48BF	0019	00000010
7	CD21	INT 21	09BD	0000	0000	0000	00FC	48BD	48AD	48BF	001B	00000010
8	58	POP AX	48BD	0000	0000	0000	00FE	48BD	48AD	48BF	001C	00000010
9	C3	RET	48BD	0000	0000	0000	0100	48BD	48AD	48BF	0028	00000010
10	BA1900	MOV DX,0019	48BD	0000	0000	0019	0100	48BD	48AD	48BF	002B	00000010
11	E8E8FF	CALL 0016	48BD	0000	0000	0019	00FE	48BD	48AD	48BF	0016	00000010
Повторяется выполнение процедуры PrintString												
12	B80700	MOV AX,0007	0007	0000	0000	0019	0100	48BD	48AD	48BF	0031	00000010
13	8AD0	MOV DL,AL	0007	0000	0000	0007	0100	48BD	48AD	48BF	0033	00000010
14	E8CAFF	CALL 0000	0007	0000	0000	0007	00FE	48BD	48AD	48BF	0000	00000010
15	80C230	ADD DL,30	0007	0000	0000	0037	00FE	48BD	48AD	48BF	0003	00000010
16	50	PUSH AX	0007	0000	0000	0037	00FC	48BD	48AD	48BF	0004	00000010
17	B402	MOV AH,02	0207	0000	0000	0037	00FC	48BD	48AD	48BF	0006	00000010
18	CD21	INT 21	0237	0000	0000	0037	00FC	48BD	48AD	48BF	0008	00000010
19	0007	POP AX	0007	0000	0000	0037	00FE	48BD	48AD	48BF	0009	00000010
20	C3	RET	0007	0000	0000	0037	0100	48BD	48AD	48BF	0036	00000010
21	E8D1FF	CALL 000A	0007	0000	0000	0037	00FE	48BD	48AD	48BF	000A	00000010
22	50	PUSH AX	0007	0000	0000	0037	00FC	48BD	48AD	48BF	000B	00000010
23	B80000	MOV AX,0000	0000	0000	0000	0037	00FC	48BD	48AD	48BF	000E	00000010
24	8AD0	MOV DL,AL	0000	0000	0000	0000	00FC	48BD	48AD	48BF	0010	00000010
25	B402	MOV AH,02	0200	0000	0000	0000	00FC	48BD	48AD	48BF	0012	00000010
26	CD21	INT 21	0200	0000	0000	0000	00FC	48BD	48AD	48BF	0014	00000010
27	58	POP AX	0007	0000	0000	0000	00FE	48BD	48AD	48BF	0015	00000010
28	C3	RET	0007	0000	0000	0000	0100	48BD	48AD	48BF	0039	00000010
29	BB0500	MOV BX,0005	0007	0005	0000	0000	0100	48BD	48AD	48BF	003C	00000010
30	003C	MOV DL,BL	0007	0005	0000	0005	0100	48BD	48AD	48BF	003E	00000010
31	E8BFFF	CALL 0000	0007	0005	0000	0005	00FE	48BD	48AD	48BF	0000	00000010
Повторяется выполнение процедуры PrintDigit												
32	BA1900	MOV DX,0019	0007	0005	0000	0019	0100	48BD	48AD	48BF	0044	00001010
33	E8CFFF	CALL 0016	0007	0005	0000	0019	00FE	48BD	48AD	48BF	0016	00001010
Повторяется выполнение процедуры PrintString												
34	93	XCHG BX,AX	0005	0007	0000	0019	0100	48BD	48AD	48BF	0048	00001010
35	8AD0	MOV DL,AL	0005	0007	0000	0005	0100	48BD	48AD	48BF	004A	00001010
36	E8B3FF	CALL 0000	0005	0007	0000	0005	00FE	48BD	48AD	48BF	0000	00001010
Повторяется выполнение процедуры PrintDigit												
37	E8BAFF	CALL 000A	0005	0007	0000	0035	00FE	48BD	48AD	48BF	000A	00001010
Повторяется выполнение процедуры PrintSpace												
38	8AD3	MOV DL,BL	0005	0007	0000	0007	0100	48BD	48AD	48BF	0052	00001010
39	E8ABFF	CALL 0000	0005	0007	0000	0007	00FE	48BD	48AD	48BF	0000	00001010
Повторяется выполнение процедуры PrintDigit												
40	B8004C	MOV AX,4C00	4C00	0007	0000	0037	0100	48BD	48AD	48BF	0058	00000010
41	CD21	INT 21	0192	000B	F715	098D	0106	2110	0192	0000	0000	10100011

5 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 регистр DL. Код используемой для вывода отдельного символа на экран Dos-функции — 02h, он помещается в регистр AH.

2. Какая операция позволяет получить для цифры её код в кодовой таблице? Если передать для отображения на экран саму цифру, то выведется символ с соответствующим ASCII-кодом, который не будет совпадать с этой цифрой. Для получения символьной формы цифры необходимо заменить цифру ASCII-кодом её изображения. Для этого нужно использовать команду ADD <регистр, содержащий цифру> 30h. 30h — ASCII-код нуля, при прибавлении к нему цифры получится ASCII-код этой цифры.

3. Объясните назначение процедуры. Как определяются начало и конец процедуры?

Современные программы обычно разрабатываются по модульному принципу — программа состоит из 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. То есть процедура нужна, чтобы выполнить небольшую часть кода, причём появляется возможность переиспользовать этот код несколько раз. Описание процедуры имеет следующий синтаксис:

<имя процедуры> PROC <параметр>

(NEAR — ближняя процедура, FAR — дальняя процедура)

<тело процедуры>

RET — возврат в основную программу

<имя процедуры> ENDP

Начало процедуры определяется с помощью директивы <название процедуры> PROC, а конец — с помощью директивы ENDP.

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

подпрограмм?

Процедуру можно разместить в любом месте программы, однако принято помещать её либо в конце сегмента кода, после команд завершения программы, либо в начале сегмента кода, до точки входа в программу, поскольку выполняться она должна только при обращении к ней. Также можно разместить её в отдельном кодовом сегменте.

5. Как процессор использует стек при работе с любой процедурой?

В процедуре ближнего типа (NEAR) при использовании команды вызова CALL указывается новое значение регистра IP, а в стеке сохраняется адрес возврата (IP команды, следующей после CALL). RET (команда косвенного перехода) же извлекает из стека одно слово и помещает его в регистр IP.

В процедуре дальнего типа (FAR) при вызове процедуры в стек кладутся значения регистров IP и CS, возврат же извлекает из стека 2 слова, слово из меньшего адреса помещается в регистр IP, а слово из большего — в CS.

6. 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 <имя процедуры>. До захода в процедуру SP был равен 0100 (стек, на который изначально выделено 256 байт, был пустым), а после вызова процедуры он стал равен 00FE, т.е. уменьшился на 2. Это связано с тем, что в стек положили значение регистра IP, соответственно указатель на вершину стека уменьшился.

7. После какой команды процедуры из стека извлекается адрес возврата? Адрес возврата из стека извлекается после команды RET, которой должна завершаться любая процедура.