

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Организация ЭВМ и систем»
Тема: Изучение режимов адресации и формирования исполнительного
адреса.

Студентка гр. 2384

Валеева А.А.

Преподаватель

Морозов С.М.

Санкт-Петербург

2023

Цель работы.

Изучить режимы адресации и формирования исполнительного адреса на языке Ассемблер.

Постановка задачи.

1. Получить у преподавателя вариант набора значений исходных данных (массивов) `vec1`, `vec2` и `matr` из файла `lr2.dat`, приведенного в каталоге Задания и занести свои данные вместо значений, указанных в приведенной ниже программе.

2. Протранслировать программу с созданием файла диагностических сообщений; объяснить обнаруженные ошибки и закомментировать соответствующие операторы в тексте программы.

3. Снова протранслировать программу и скомпоновать загрузочный модуль.

4. Выполнить программу в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения команды.

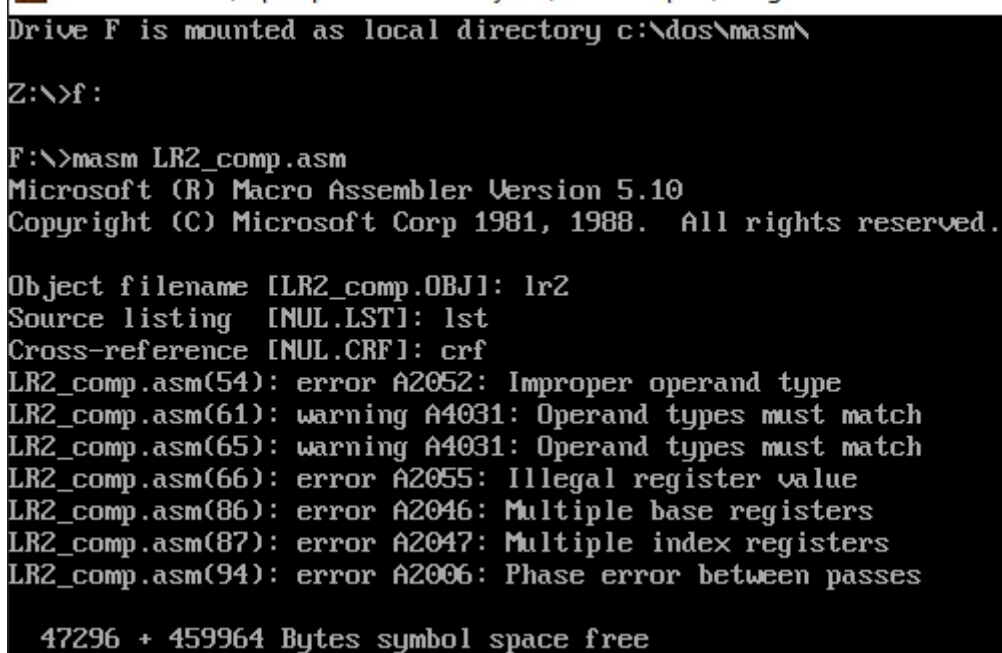
5. Результаты прогона программы под управлением отладчика должны быть подписаны преподавателем и представлены в отчете.

Выполнение работы.

1. Был загружен файл `LR2_comp.ASM` в каталог с компилятором `MASM`
2. Исходный код программы был просмотрен в режиме редактирования с помощью редактора *Geany*. Была изучена структура и реализация каждого сегмента программы.
3. Был изменён набор значений в файле `LR2_comp.ASM`, заменяющих приведённые в образце программы.

4. Трансляция исходного кода.

- 1) Открыть программу DOSBOX.
- 2) Транслируем программу в DOSBOX следующим образом:
монтируем директорию с помощью команды «**mount f**
c:\dos\MASM» и совершаем переход в нее «**f:**». Транслируем
программу с помощью команды «**masm LR2_comp.ASM**»



```
Drive F is mounted as local directory c:\dos\masm\  
Z:\>f:  
F:\>masm LR2_comp.asm  
Microsoft (R) Macro Assembler Version 5.10  
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.  
  
Object filename [LR2_comp.OBJ]: lr2  
Source listing [NUL.LST]: lst  
Cross-reference [NUL.CRF]: crf  
LR2_comp.asm(54): error A2052: Improper operand type  
LR2_comp.asm(61): warning A4031: Operand types must match  
LR2_comp.asm(65): warning A4031: Operand types must match  
LR2_comp.asm(66): error A2055: Illegal register value  
LR2_comp.asm(86): error A2046: Multiple base registers  
LR2_comp.asm(87): error A2047: Multiple index registers  
LR2_comp.asm(94): error A2006: Phase error between passes  
  
47296 + 459964 Bytes symbol space free
```

В процессе трансляции был создан файл листинга `lst.lst`. Файл листинга содержит диагностическую информацию в виде сообщений о двух предупреждениях (Warning errors) и пяти ошибках (Severe errors).

- 3) Демонстрация кода файла листинга приведена в приложении А.
Ошибки и предупреждения выделены жирным.

Объяснение ошибок.

1. LR2_comp-asm(54): error A2052: Improper operand type - данная ошибка возникает из-за строки `mov mem3, [bx]` инструкция `mov` неспособна перенести значение из одной ячейки памяти в другую. (нужно использовать промежуточное значение)

2. LR2_comp.asm(61): warning A4031: Operand types must match – данную ошибку вызывает часть код `mov cx, vec2[di]`, так как мы пытаемся положить данные из ячейки памяти с размером 1 байт в регистр с размером 2 байта.
 3. LR2_comp.asm(65): warning A4031: Operand types must match – из-за строки `mov cx, matr[bx][di]`, пытаемся положить данные из ячейки памяти с размером 1 байт в регистр с размером 2 байта.
 4. LR2_comp.asm(66): error A2055: Illegal register value: Illegal register value часть кода: `mov ax, matr[bx*4][di]`. Регистр недопустимый, т.к. масштабирование применимо только к 32-битным регистрам, а здесь используется базово-индексная адресация. В данном случае необходимо сначала изменить значение регистра, а потом переводить информацию.
 5. LR2_comp.asm(86): error A2046: Multiple base registers – `mov ax, matr[bp+bx]` – недопустимо использовать несколько базовых регистров для адресации.
 6. LR2_comp-asm(87): error A2047: Multiple index registers – `mov ax, matr[bp+di+si]` – недопустимо использовать несколько индексных регистров для адресации.
 7. LR2_comp.asm(94): error A2006: Phase error between passes – внутренняя ошибка компилятора вызванная ошибками в других строках программы (пропадает после исправления остальных ошибок).
5. В режиме редактирования были закомментированы строки с ошибками, строки с предупреждениями остались без изменений. В

DOSBox повторно введена команда **masm LR2_comp.asm**. Был создан объектный файл **LR2_COMP.obj** и листинг **lr2.lst**. Осталось только два предупреждения, ошибки не возникли. Код программы можно посмотреть в приложении Б.

```
F:\>masm LR2_comp.asm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

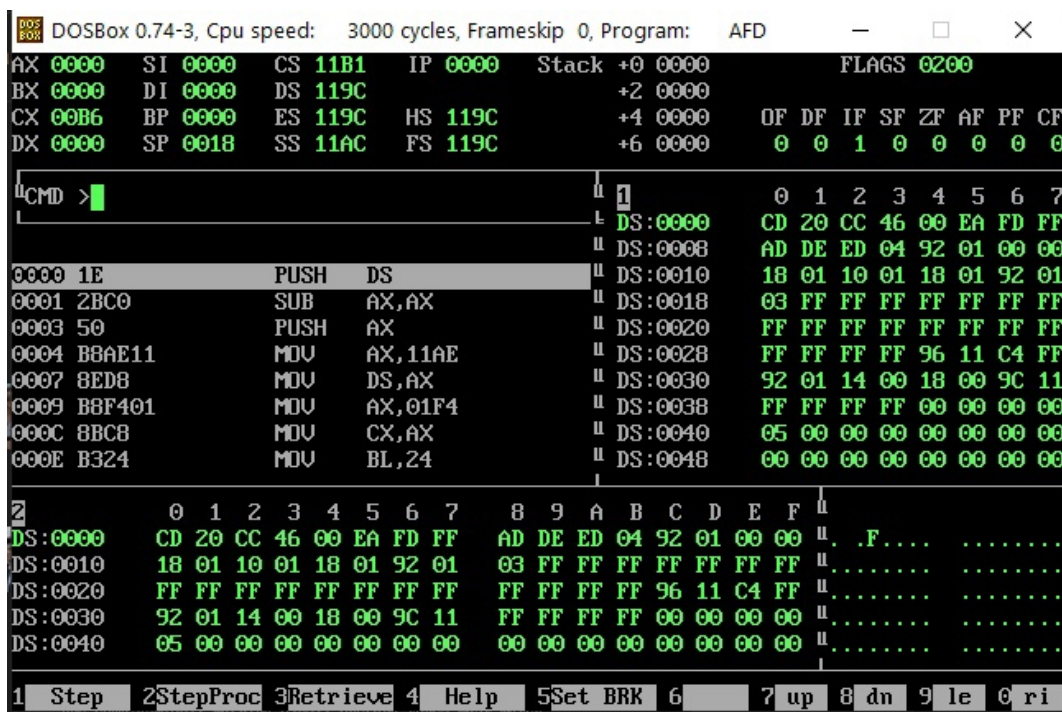
Object filename [LR2_comp.OBJ]:
Source listing [NUL.LST]: LR2.lst
Cross-reference [NUL.CRF]:
LR2_comp.asm(60): warning A4031: Operand types must match
LR2_comp.asm(64): warning A4031: Operand types must match

47812 + 459448 Bytes symbol space free

2 Warning Errors
0 Severe Errors

F:\>
```

- Командой **link.exe LR2_comp.obj** был скомпонован **LR2_comp.exe** и создан **LR2.map**. Программа была выполнена в режиме отладки командой **afd LR2_comp.exe**.



Содержимое регистров до выполнения прогона программы: AX: 0000, BX: 0000, CX: 00B6, DX: 0000, CS: 11B1, DS: 119C, ES: 119C, SS: 11AC, IP: 0000

- Результат работы программы.

Адрес команды	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(IP) = 0000 (SP)=0018 Stack +0 0000 +2 0000 +4 0000 +6 0000	(IP) = 0001 (SP)=0016 Stack +0 119C +2 0000 +4 0000 +6 0000
0001	SUB AX, AX	2BC0	(IP) = 0001 (AX) = 0000	(IP) = 0003 (AX) = 0000
0003	PUSH AX	50	(IP) = 0003 (SP)=0016 Stack +0 119C +2 0000 +4 0000 +6 0000	(IP) = 0004 (SP)=0014 Stack +0 0000 +2 119C +4 0000 +6 0000
0004	MOV AX, 11AE	B8AE11	(IP) = 0004 (AX) = 0000	(IP) = 0007 (AX) = 11AE
0007	MOV DS, AX	8ED8	(IP) = 0007 (DS) = 119C	(IP) = 0009 (DS) = 11AE
0009	MOV AX, 01F4	B8F401	(IP) = 0009 (AX) = 11AE	(IP) = 000C (AX) = 01F4
000C	MOV CX, AX	8BC8	(IP) = 000C (CX)=00B6	(IP) = 000E (CX)=01F4

000E	MOV BL, 24	B324	(IP) = 000E (BX) = 0000	(IP) = 0010 (BX) = 0024
0010	MOV BH, CE	B7CE	(IP) = 0010 (BX) = 0024	(IP) = 0012 (BX) = CE24
0012	MOV [0002], FFCE	C7060200C EFF	(IP) = 0012 DS:0002 00 DS:0003 00	(IP) = 0018 DS:0002 CE DS:0003 FF
0018	MOV BX, 0006	BB0600	(IP) = 0018 (BX) = CE24	(IP) = 001B (BX) = 0006
001B	MOV [0000], AX	A30000	(IP) = 001B DS:0000 00 DS:0001 00	(IP) = 001E DS:0000 F4 DS:0001 01
001E	MOV AL, [BX]	8A07	(IP) = 001E (AX) = 01F4	(IP) = 0020 (AX) = 010C
0020	MOV AL, [BX+03]	8A4703	(IP) = 0020 (AX) = 010C	(IP) = 0023 (AX) = 0109
0023	MOV CX, [BX+03]	8B4F03	(IP) = 0023 (CX) = 01F4	(IP) = 0026 (CX) = 0509
0026	MOV DI, 0002	BF0200	(IP) = 0026 (DI) = 0000	(IP) = 0029 (DI) = 0002
0029	MOV AL, [DI+000E]	8A850E00	(IP) = 0029 (AX) = 0109	(IP) = 002D (AX) = 01D8
002D	MOV CX, [DI+000E]	8B8D0E00	(IP) = 002D (CX) = 0509	(IP) = 0031 (CX) = CED8
0031	MOV BX, 0003	BB0300	(IP) = 0031 (BX) = 0006	(IP) = 0034 (BX) = 0003

0034	MOV AL, [BX+DI+0016]	8A811600	(IP) = 0034 (AX) = 01D8	(IP) = 0038 (AX) = 01F9
0038	MOV CX, [BX+DI+0016]	8B891600	(IP)=0038 (CX)=CED8	(IP)=003C (CX)=FAF9
003C	MOV AX, 11AE	B8AE11	(IP) = 003C (AX) = 01F9	(IP) = 003F (AX) = 11AE
003F	MOV ES, AX	8ECO	(IP) = 003F (ES) = 119C	(IP) = 0041 (ES) = 11AE
0041	MOV AX, ES:[BX]	268B07	(IP) = 0041 (AX) = 11AE	(IP) = 0044 (AX) = 00FF
0044	MOV AX, 0000	B80000	(IP) = 0044 (AX) = 00FF	(IP) = 0047 (AX) = 0000
0047	MOVE ES, AX	8ECO	(IP) = 0047 (ES) = 11AE	(IP) = 0049 (ES) = 0000
0049	PUSH DS	1E	(IP) = 0049 (SP) = 0014 Stack +0 0000 +2 119C +4 0000 +6 0000	(IP) = 004A (SP) = 0012 Stack +0 11AE +2 0000 +4 119C +6 0000
004A	POP ES	07	(IP) = 004A (SP) = 0012 Stack +0 11AE +2 0000 +4 119C	(IP) = 004B (SP) = 0014 Stack +0 0000 +2 119C +4 0000

			+6 0000	+6 0000
004B	MOV CX, ES:[BX-01]	268B4FFF	(IP) = 004B (CX) = FAF9	(IP) = 004F (CX) = FFCE
004F	XCHG	91	(IP) = 004F (AX) = 0000 (CX) = FFCE	(IP) = 0050 (AX) = FFCE (CX) = 0000
0050	MOV DI, 0002	BF0200	(IP) = 0050 (DI) = 0002	(IP) = 0053 (DI) = 0002
0053	MOV ES:[BX+DI], AX	268901	(IP) = 0053 DS:0005 00 DS:0006 0C	(IP) = 0056 DS:0005 CE DS:0006 FF
0056	MOV BP, SP	8BEC	(IP) = 0056 (BP) = 0000	(IP) = 0058 (BP) = 0014
0058	PUSH [0000]	FF360000	(IP) = 0058 (SP) = 0014 Stack +0 0000 +2 119C +4 0000 +6 0000	(IP) = 005C (SP) = 0012 Stack +0 01F4 +2 0000 +4 119C +6 0000
005C	PUSH [0002]	FF360200	(IP) = 005C (SP) = 0012 Stack +0 01F4 +2 0000 +4 119C +6 0000	(IP) = 0060 (SP) = 0010 Stack +0 FFCE +2 01F4 +4 0000 +6 119C

0060	MOV BP, SP	8BEC	(IP) = 0060 (BP) = 0014	(IP) = 0062 (BP) = 0010
0062	MOV DX, [BP+02]	8B5602	(IP) = 0062 (DX) = 0000	(IP) = 0065 (DX) = 01F4
0065	RET FAR	CB	(IP) = 0065 (SP) = 0010 (CS) = 11B1 Stack +0 FFCE +2 01F4 +4 0000 +6 119C	(IP) = FFCE (SP) = 0014 (CS) = 01F4 Stack +0 0000 +2 119C +4 0000 +6 0000

Вывод.

В результате выполнения лабораторной работы были изучены различные виды адресации (регистровая, прямая, косвенная, базированная, индексированная адресации и адресация с базированием и индексированием).

ФАЙЛЫ С ОШИБКАМИ

Файл: LR2_comp.asm

```
; Учебная программа лабораторной работы №2 по дисциплине "Организация ЭВМ и С";
;
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50

; Стек программы

AStack SEGMENT STACK
        DW 12 DUP(?)
AStack ENDS

; Данные программы

DATA SEGMENT

; Директивы описания данных
```

```

mem1      DW      0
mem2      DW      0
mem3      DW      0
vec1      DB      12,11,10,9,5,6,7,8
vec2      DB      40,50,-40,-50,-20,-30,20,30
matr      DB      5,6,7,8,-8,-7,-6,-5,1,2,3,4,-4,-3,-2,-1

```

```
DATA      ENDS
```

```
; Код программы
```

```
CODE      SEGMENT
          ASSUME CS:CODE, DS:DATA, SS:AStack
```

```
; Головная процедура
```

```
Main     PROC    FAR
          push    DS
          sub     AX,AX
          push    AX
          mov     AX,DATA
          mov     DS,AX
```

```
; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИЙ
```

```
; Регистровая адресация
```

```
          mov     ax,n1
          mov     cx,ax
          mov     bl,EOL
          mov     bh,n2
```

```
; Прямая адресация
```

```
          mov     mem2,n2
          mov     bx,OFFSET vec1
          mov     mem1,ax
```

```
; Косвенная адресация
```

```
          mov     al,[bx]
          mov     mem3,[bx]
```

```
; Базированная адресация
```

```
          mov     al,[bx]+3
          mov     cx,3[bx]
```

```
; Индексированная адресация
```

```
          mov     di,ind
          mov     al,vec2[di]
          mov     cx,vec2[di]
```

```
; Адресация с базированием и индексированием
```

```
          mov     bx,3
          mov     al,matr[bx][di]
          mov     cx,matr[bx][di]
          mov     ax,matr[bx*4][di]
```

```
; ПРОВЕРКА АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
```

```
; Переопределение сегмента
```

```
; ----- вариант 1
```

```
          mov     ax, SEG vec2
          mov     es, ax
          mov     ax, es:[bx]
          mov     ax, 0
```

```
; ----- вариант 2
```

```
          mov     es, ax
          push    ds
          pop     es
          mov     cx, es:[bx-1]
          xchg    cx,ax
```

```
; ----- вариант 3
```

```

        mov  di,ind
        mov  es:[bx+di],ax
;  ----- вариант 4
        mov  bp,sp
        mov  ax,matr[bp+bx]
        mov  ax,matr[bp+di+si]
;  Использование сегмента стека
        push mem1
        push mem2
        mov  bp,sp
        mov  dx,[bp]+2
        ret
Main     ENDP
CODE     ENDS
        END Main

```

Файл: LST.LST

Microsoft (R) Macro Assembler Version 5.10
12:14:08

10/8/23

Page

1-1

```

; Учебная программа лабораторной работы №2 по
; дисциплине "Организация ЭВМ и С";
;
= 0024          EOL   EQU   '$'
= 0002          ind   EQU   2
= 01F4          n1    EQU   500
=-0032          n2    EQU   -50

; Стек программы

0000          AStack  SEGMENT  STACK
0000  000C[      DW  12 DUP(?)
      ????
      ]

0018          AStack  ENDS

; Данные программы

0000          DATA    SEGMENT

; Директивы описания данных

0000  0000          mem1      DW      0
0002  0000          mem2      DW      0
0004  0000          mem3      DW      0
0006  01 02 03 04 08 07 vec1   DB      1,2,3,4,8,7,6,5
      06 05

```

```

000E  F6 EC 0A 14 E2 D8 vec2      DB    -10,-20,10,20,-30,-40,30,40
      1E 28
0016  01 02 03 04 FC FD matr      DB          1,2,3,4,-4,-3,-2,-
1,5,6,7,8,-8,
      -7,-6,-5
      FE FF 05 06 07 08
      F8 F9 FA FB

0026                                     DATA      ENDS

; Код программы

0000                                     CODE      SEGMENT
                                         ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
0000                                     Main      PROC   FAR
0000  1E                                     push   DS
0001  2B C0                                     sub     AX,AX
0003  50                                     push   AX
0004  B8 ---- R                               mov     AX,DATA
0007  8E D8                                     mov     DS,AX

;   ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИ
;   Й
;   Регистровая адресация
0009  B8 01F4                               mov     ax,n1

```

1-2

```
000C  8B C8                      mov  cx,ax
000E  B3 24                      mov  bl,EOL
0010  B7 CE                      mov  bh,n2
                                ; Прямая адресация
0012  C7 06 0002 R FFCE        mov  mem2,n2
0018  BB 0006 R                mov  bx,OFFSET vec1
001B  A3 0000 R                mov  mem1,ax
                                ; Косвенная адресация
001E  8A 07                      mov  al,[bx]
                                mov  mem3,[bx]
LR2.asm(54): error A2052: Improper operand type
                                ; Базированная адресация
0020  8A 47 03                  mov  al,[bx]+3
0023  8B 4F 03                  mov  cx,3[bx]
                                ; Индексированная адресация
0026  BF 0002                  mov  di,ind
0029  8A 85 000E R            mov  al,vec2[di]
002D  8B 8D 000E R            mov  cx,vec2[di]
LR2.asm(61): warning A4031: Operand types must match
                                ; Адресация с базированием и индексированием
0031  BB 0003                  mov  bx,3
0034  8A 81 0016 R            mov  al,matr[bx][di]
0038  8B 89 0016 R            mov  cx,matr[bx][di]
LR2.asm(65): warning A4031: Operand types must match
003C  8B 85 0022 R            mov  ax,matr[bx*4][di]
LR2.asm(66): error A2055: Illegal register value

                                ; ПРОВЕРКА АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
                                ; Переопределение сегмента
                                ; ----- вариант 1
0040  B8 ---- R                mov  ax, SEG vec2
0043  8E C0                      mov  es, ax
0045  26: 8B 07                  mov  ax, es:[bx]
0048  B8 0000                    mov  ax, 0
                                ; ----- вариант 2
004B  8E C0                      mov  es, ax
004D  1E                          push ds
004E  07                          pop  es
004F  26: 8B 4F FF              mov  cx, es:[bx-1]
0053  91                          xchg cx,ax
                                ; ----- вариант 3
0054  BF 0002                  mov  di,ind
0057  26: 89 01                  mov  es:[bx+di],ax
                                ; ----- вариант 4
005A  8B EC                      mov  bp,sp
005C  3E: 8B 86 0016 R          mov  ax,matr[bp+bx]
LR2.asm(86): error A2046: Multiple base registers
```

```

0061 3E: 8B 83 0016 R          mov ax,matr[bp+di+si]
LR2.asm(87): error A2047: Multiple index registers
          ; Использование сегмента стека
0066 FF 36 0000 R          push mem1
006A FF 36 0002 R          push mem2
006E 8B EC          mov bp,sp
0070 8B 56 02          mov dx,[bp]+2
0073 CB          ret
0074          Main          ENDP
LR2.asm(94): error A2006: Phase error between passes
0074          CODE          ENDS
          END Main
Microsoft (R) Macro Assembler Version 5.10          10/8/23
12:14:08

```

Symbols-1

Segments and Groups:

Class	N a m e	Length	Align	Combine
ASTACK	0018	PARA	STACK
CODE	0074	PARA	NONE
DATA	0026	PARA	NONE

Symbols:

	N a m e	Type	Value	Attr
EOL	NUMBER	0024	
IND	NUMBER	0002	
MAIN	F PROC	0000	CODE Length =
	0074			
MATR	L BYTE	0016	DATA
MEM1	L WORD	0000	DATA
MEM2	L WORD	0002	DATA
MEM3	L WORD	0004	DATA
N1	NUMBER	01F4	
N2	NUMBER	-0032	
VEC1	L BYTE	0006	DATA
VEC2	L BYTE	000E	DATA
@CPU	TEXT	0101h	
@FILENAME	TEXT	LR2	
@VERSION	TEXT	510	

96 Source Lines
96 Total Lines
19 Symbols

47842 + 459418 Bytes symbol space free

2 Warning Errors
5 Severe Errors

ПРИЛОЖЕНИЕ Б

ФАЙЛЫ ИСПРАВЛЕННОЙ ПРОГРАММЫ

ФАЙЛ: LR2_COMP.ASM

```
; Учебная программа лабораторной работы №2 по дисциплине "Организация ЭВМ
и С";
;
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50

; Стек программы

AStack SEGMENT STACK
        DW 12 DUP(?)
AStack ENDS

; Данные программы

DATA SEGMENT

; Директивы описания данных

mem1 DW 0
mem2 DW 0
mem3 DW 0
vec1 DB 12,11,10,9,5,6,7,8
vec2 DB 40,50,-40,-50,-20,-30,20,30
matr DB 5,6,7,8,-8,-7,-6,-5,1,2,3,4,-4,-3,-2,-1

DATA ENDS

; Код программы

CODE SEGMENT
        ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
Main PROC FAR
        push DS
        sub AX,AX
        push AX
```



```

        mov     AX, DATA
        mov     DS, AX

;   ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИЙ
;   Регистровая адресация
        mov     ax, n1
        mov     cx, ax
        mov     bl, EOL
        mov     bh, n2
;   Прямая адресация
        mov     mem2, n2
        mov     bx, OFFSET vec1
        mov     mem1, ax
;   Косвенная адресация
        mov     al, [bx]
;   Базированная адресация
        mov     al, [bx]+3
        mov     cx, 3[bx]
;   Индексированная адресация
        mov     di, ind
        mov     al, vec2[di]
        mov     cx, vec2[di]
;   Адресация с базированием и индексированием
        mov     bx, 3
        mov     al, matr[bx][di]
        mov     cx, matr[bx][di]

;   ПРОВЕРКА АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
;   Переопределение сегмента
;   ----- вариант 1
        mov     ax, SEG vec2
        mov     es, ax
        mov     ax, es:[bx]
        mov     ax, 0
;   ----- вариант 2
        mov     es, ax
        push    ds
        pop     es
        mov     cx, es:[bx-1]
        xchg    cx, ax
;   ----- вариант 3
        mov     di, ind
        mov     es:[bx+di], ax
;   ----- вариант 4
        mov     bp, sp
;   Использование сегмента стека
        push    mem1
        push    mem2
        mov     bp, sp
        mov     dx, [bp]+2
        ret
Main     ENDP
CODE     ENDS
        END Main

```

Файл: LST_2.LST

Microsoft (R) Macro Assembler Version 5.10

12/10/21 17:11:2

Page 1-1

```
; Учебная программа лабораторной работы №2 по
дисциплине "Архитектура компьютера"
;
;
= 0024          EOL   EQU   '$'
= 0002          ind   EQU   2
= 01F4          n1    EQU   500
=-0032          n2    EQU   -50

; Стек программы

0000          AStack      SEGMENT  STACK
0000  000C[          DW 12 DUP(?)
          ????
          ]

0018          AStack      ENDS

; Данные программы

0000          DATA      SEGMENT

; Директивы описания данных

0000  0000          mem1      DW      0
0002  0000          mem2      DW      0
0004  0000          mem3      DW      0
0006  12 11 10 0F 0B 0C          vec1      DB      18,17,16,15,11,12,13,14
          0D 0E
000E  1E 28 E2 D8 0A 14          vec2      DB      30,40,-30,-40,10,20,-10,-20
          F6 EC
0016  FC FD 01 02 FE FF          matr      DB      -4,-3,1,2,-2,-1,3,4,5,6,7,8,-
8,
          -7,-6,-5
          03 04 05 06 07 08
          F8 F9 FA FB

0026          DATA      ENDS

; Код программы

0000          CODE      SEGMENT
          ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
0000  Main          PROC   FAR
0000  1E          push    DS
0001  2B C0          sub     AX,AX
0003  50          push    AX
0004  B8 ---- R      mov     AX,DATA
0007  8E D8          mov     DS,AX

; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИ
Й
```

```

0009 B8 01F4          mov ax,n1
000C 8B C8          mov cx,ax
000E B3 24          mov bl,EOL
0010 B7 CE          mov bh,n2
; Прямая адресация
0012 C7 06 0002 R FFCE  mov mem2,n2
0018 BB 0006 R      mov bx,OFFSET vec1
001B A3 0000 R      mov mem1,ax
; Косвенная адресация
001E 8A 07          mov al,[bx]
; mov mem3,[bx]
; Базированная адресация
0020 8A 47 03      mov al,[bx]+3
0023 8B 4F 03      mov cx,3[bx]
; Индексированная адресация
0026 BF 0002      mov di,ind
0029 8A 85 000E R  mov al,vec2[di]
002D 8B 8D 000E R  mov cx,vec2[di]
LR2_COMP.ASM(62): warning A4031: Operand types must match
; Адресация с базированием и индексированием
0031 BB 0003      mov bx,3
0034 8A 81 0016 R  mov al,matr[bx][di]
0038 8B 89 0016 R  mov cx,matr[bx][di]
LR2_COMP.ASM(66): warning A4031: Operand types must match
; mov ax,matr[bx*4][di]
; ПРОВЕРКА АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
; Переопределение сегмента
; ----- вариант 1
003C B8 ---- R      mov ax, SEG vec2
003F 8E C0          mov es, ax
0041 26: 8B 07      mov ax, es:[bx]
0044 B8 0000          mov ax, 0
; ----- вариант 2
0047 8E C0          mov es, ax
0049 1E            push ds
004A 07            pop es
004B 26: 8B 4F FF      mov cx, es:[bx-1]
004F 91            xchg cx,ax
; ----- вариант 3
0050 BF 0002      mov di,ind
0053 26: 89 01      mov es:[bx+di],ax
; ----- вариант 4
0056 8B EC          mov bp,sp
; mov ax,matr[bp+bx]
; mov ax,matr[bp+di+si]
; Использование сегмента стека
0058 FF 36 0000 R  push mem1
005C FF 36 0002 R  push mem2
0060 8B EC          mov bp,sp
0062 8B 56 02      mov dx,[bp]+2
0065 CB            ret
0066 Main          ENDP
0066 CODE          ENDS
; END Main

```

Segments and Groups:

N a m e	Length	Align	Combine Class
ASTACK	0018	PARA	STACK
CODE	0066	PARA	NONE
DATA	0026	PARA	NONE

Symbols:

N a m e	Type	Value	Attr
EOL	NUMBER	0024	
IND	NUMBER	0002	
MAIN	F PROC	0000	CODE Length = 0066
MATR	L BYTE	0016	DATA
MEM1	L WORD	0000	DATA
MEM2	L WORD	0002	DATA
MEM3	L WORD	0004	DATA
N1	NUMBER	01F4	
N2	NUMBER	-0032	
VEC1	L BYTE	0006	DATA
VEC2	L BYTE	000E	DATA
@CPU	TEXT	0101h	
@FILENAME	TEXT	LR2_COMP	
@VERSION	TEXT	510	

97 Source Lines

97 Total Lines

19 Symbols

47806 + 459454 Bytes symbol space free

2 Warning Errors

0 Severe Errors

Файл: MAP.MAP

Start	Stop	Length	Name	Class
00000H	00017H	00018H	ASTACK	
00020H	00045H	00026H	DATA	
00050H	000B5H	00066H	CODE	

Program entry point at 0005:0000