

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МО ЭВМ**

**ОТЧЕТ**  
**по лабораторной работе №2**  
**по дисциплине «Организация ЭВМ и систем»**  
**Тема: Изучение режимов адресации и формирования**  
**исполнительного адреса**  
**Вариант 3**

Студентка гр. 0382

\_\_\_\_\_

Деткова А.С.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2021

### **Цель работы.**

Изучить режимы адресации памяти и формирования исполнительного адреса в архитектуре Intel X86.

### **Задание.**

Вариант 3.

Лабораторная работа 2 предназначена для изучения режимов адресации, использует готовую программу `lr2_comp.asm` на Ассемблере, которая в автоматическом режиме выполняться не должна, так как не имеет самостоятельного функционального назначения, а только тестирует режимы адресации. Поэтому ее выполнение должно производиться под управлением отладчика в пошаговом режиме.

В программу введен ряд ошибок, которые необходимо объяснить в отчете по работе, а соответствующие команды закомментировать для прохождения трансляции. Необходимо составить протокол выполнения программы в пошаговом режиме отладчика по типу таблицы 1 предыдущей лабораторной работы и подписать его у преподавателя. На защите студенты должны уметь объяснить результат выполнения каждой команды с учетом используемого вида адресации. Результаты, полученные с помощью отладчика, не являются объяснением, а только должны подтверждать ваши объяснения.

### **Порядок выполнения работы.**

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.

Данные для варианта 3 представлены на рисунке 1.

3	vec1	8,7,6,5,1,2,3,4
	vec2	-30,-40,30,40,-10,-20,10,20
	matr	-1,-2,-3,-4,8,7,6,5,-5,-6,-7,-8,4,3,2,1

Рисунок 1: Данные варианта 3

Файл диагностических сообщений с ошибками и после исправления см. в приложении Б. Исходный программный код с исправленными ошибками см. в приложении А.

Все ошибки с объяснениями причин представлены в таблице 1.

Таблица 1: Ошибки в программе с объяснением

Ошибки	Объяснение
mov mem3,[bx]	Чтение из памяти и одновременная запись в память запрещена на архитектурном уровне.
mov cx,vec2[di]	Размер приемника и источника должны быть одинаковы. Попытка поместить данные размером 1 байт в источник размером 2 байта.
mov cx,matr[bx][di]	Размер приемника и источника должны быть одинаковы. Попытка поместить данные размером 1 байт в источник

	размером 2 байта.
mov ax,matr[bx*4][di]	Запрещено масштабирование по 2х байтовым регистрам.
mov ax,matr[bp+bx]	Запрещено использование нескольких регистров общего назначения для адресации .
mov ax,matr[bp+di+si]	Запрещено использование нескольких индексных регистров для адресации.

Результат пошагового выполнения программы представлен в таблице 2.

Начальные значения сегментных регистров:

(CS) = 1A0A; (DS) = 19F5; (ES) = 19F5; (SS) = 1A05

Таблица 2: Протокол отладки программы

Адрес с командами	Символический код команды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(DS) = 19F5 (IP) = 0000 (SP) = 0018 Stack +0 0000 +2 0000	(DS) = 19F5 (IP) = 0001 (SP) = 0016 Stack +0 19F5 +2 0000
0001	SUB AX,AX	2BC0	(AX) = 0000 (IP) = 0001	(AX) = 0000 (IP) = 0003
0003	PUSH AX	50	(IP) = 0003 (SP) = 0016 Stack +0 19F5 +2 0000	(IP) = 0004 (SP) = 0014 Stack +0 0000 +2 19F5
0004	MOV AX,1A07	B8071A	(AX) = 0000 (IP) = 0004	(AX) = 1A07 (IP) = 0007
0007	MOV DS,AX	8ED8	(DS) = 19F5 (IP) = 0007	(DS) = 1A07 (IP) = 0009
0009	MOV AX,01F4	B8F401	(AX) = 1A07 (IP) = 0009	(AX) = 01F4 (IP) = 000C

000C	MOV CX,AX	8BC8	(AX) = 01F4 (CX) = 00B0 (IP) = 000C	(AX) = 01F4 (CX) = 01F4 (IP) = 000E
000E	MOV BL, 24	B324	(BX) = 0000 (IP) = 000E	(BX) = 0024 (IP) = 0010
0010	MOV BH,CE	B7CE	(BX) = 0024 (IP) = 0010	(BX) = CE24 (IP) = 0012
0012	MOV [0002],FFCE	C7060200CE FF	DS:0002 = 00 DS:0003 = 00 (IP) = 0012	DS:0002 = CE DS:0003 = FF (IP) = 0018
0018	MOV BX,0006	BB0600	(BX) = CE24 (IP) = 0018	(BX) = 0006 (IP) = 001B
001B	MOV [0000],AX	A30000	DS:0000 = 00 DS:0001 = 00 (IP) = 001B	DS:0000 = F4 DS:0001 = 01 (IP) = 001E
001E	MOV AL,[BX]	8A07	(AX) = 01F4 [BX] = DS:0006 = 08 (IP) = 001E	(AX) = 0108 (IP) = 0020 [BX] = DS:0006 = 08
0020	MOV AL,[BX+03]	8A4703	(AX) = 0108 [BX+03]=DS:0009=0 5 (IP) = 0020	(AX) = 0105 [BX+03]=DS:0009=0 5 (IP) = 0023
0023	MOV CX,[BX+03]	8B4F03	(CX) = 01F4 [BX+03] = DS:0009 = 05 (IP) = 0023	(CX) = 0105 [BX+03] = DS:0009 = 05 (IP) = 0026
0026	MOV DI,0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL,[000E+DI]	8A50E00	(AX) = 0105 [000E+DI] = DS:0010 = 1E (IP) = 0029	(AX) = 011E [000E+DI] = DS:0010 = 1E (IP) = 002D
002D	MOV BX,0003	BB0300	(BX) = 0006 (IP) = 002D	(BX) = 0003 (IP) = 0030

0030	MOV AL, [0016+BX+DI]	8A811600	[0016+BX+DI] = DS:001B = 07 (AX) = 011E (IP) = 0030	[0016+BX+DI] = DS:001B = 07 (AX) = 0107 (IP) = 0034
0034	MOV AX,1A07	B8071A	(AX) = 0107 (IP) = 0034	(AX) = 1A07 (IP) = 0037
0037	MOV ES,AX	8EC0	(ES) = 19F5 (IP) = 0037	(ES) = 1A07 (IP) = 0039
0039	MOV AX,ES:[BX]	268B07	(AX) = 1A07 ES:[BX] = ES:0003 = 00FF (IP) = 0039	(AX) = 00FF ES:[BX] = ES:0003 = 00FF (IP) = 003C
003C	MOV AX,0000	B80000	(AX) = 00FF (IP) = 003C	(AX) = 0000 (IP) = 003F
003F	MOV ES,AX	8EC0	(ES) = 1A07 (IP) = 003F	(ES) = 0000 (IP) = 0041
0041	PUSH DS	1E	(DS) = 1A07 (IP) = 0041 Stack +0 0000 +2 19F5 +4 0000	(DS) = 1A07 (IP) = 0042 Stack +0 1A07 +2 0000 +4 19F5
0042	POP ES	07	(ES) = 0000 (IP) = 0042 Stack +0 1A07 +2 0000 +4 19F5	(ES) = 1A07 (IP) = 0043 Stack +0 0000 +2 19F5
0043	MOV CX,ES:[BX-01]	268B4FFF	(CX) = 0105 ES:[BX-01] = DS:0002 = FFCE (IP) = 0043	(CX) = FFCE ES:[BX-01] = DS:0002 = FFCE (IP) = 0047
0047	XCHG AX,CX	91	(CX) = FFCE (AX) = 0000 (IP) = 0047	(CX) = 0000 (AX) = FFCE (IP) = 0047
0048	MOV DI, 0002	BF0200	(DI) = 0002	(DI) = 0002

			(IP) = 0048	(IP) = 004B
004B	MOV ES:[BX+DI],AX	268901	ES:[BX+DI] = DS: [0003+0002] = DS: [0005] = 0800 (AX) = FFCE (IP) = 004B	ES:[BX+DI] = DS: [0003+0002] = DS: [0005] = FFCE (AX) = FFCE (IP) = 004E
004E	MOV BP,SP	8BEC	(BP) = 0000 (IP) = 004E	(BP) = 0014 (IP) = 0050
0050	PUSH [0000]	FF360000	[0000] = 01F4 (IP) = 0050 (SP) = 0014 Stack +0 0000 +2 19F5	[0000] = 01F4 (IP) = 0054 (SP) = 0012 Stack +0 01F4 +2 0000 +4 19F5
0054	PUSH [0002]	FF360200	[0002] = FFCE (IP) = 0054 (SP) = 0012 Stack +0 01F4 +2 0000 +4 19F5	[0002] = FFCE (IP) = 0058 (SP) = 0010 Stack +0 FFCE +2 01F4 +4 0000 +6 19F5
0058	MOV BP,SP	8BEC	(BP) = 0014 (IP) = 0058	(BP) = 0010 (IP) = 005A
005A	MOV DX,[BP+02]	8B5602	(DX) = 0000 [BP+02] = 01F4 (IP) = 005A	(DX) = 01F4 [BP+02] = 01F4 (IP) = 005D
005D	RET Far 0002	CA0200	(IP) = 005D (SP) = 0010 (CS) = 1A0A Stack +0 FFCE +2 01F4 +4 0000 +6 19F5	(IP) = FFCE (SP) = 0016 (CS) = 01F4 Stack +0 19F5

### **Выводы.**

Были изучены режимы адресации памяти и формирования исполнительного адреса в архитектуре Intel X86.



## ПРИЛОЖЕНИЕ А

### ИСПРАВЛЕННЫЙ КОД ПРОГРАММЫ

Название файла: LB2.asm

; Программа изучения режимов адресации процессора IntelX86

```
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 8,7,6,5,1,2,3,4  
vec2 DB -30,-40,30,40,-10,-20,10,20  
matr DB -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-8,4,3,2,1  
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]

; Индексная 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 РЕЖИМОВ 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 2
Main ENDP
CODE ENDS
    END Main

```

## ПРИЛОЖЕНИЕ Б

### ЛИСТИНГИ ПРОГРАММЫ

Название файла: LB2.LST

Microsoft (R) Macro Assembler Version 5.10

10/3/21

23:36:31

Page 1-1

; Программа изучения режимов адресации процессора IntelX86

```
= 0024          EOL EQU '$'
= 0002          ind EQU 2
= 01F4          n1 EQU 500
=-0032          n2 EQU -50
```

; Стек программы

```
0000          AStack SEGMENT STACK
0000 0000C[          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 08 07 06 05 01 02  vec1 DB 8,7,6,5,1,2,3,4
          03 04
000E E2 D8 1E 28 F6 EC  vec2 DB -30,-40,30,40,-10,-20,10,20
          0A 14
0016 FF FE FD FC 08 07  matr DB -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-
8,4,3,2,1
```

```

0026          06 05 FB FA F9 F8          04 03 02 01
          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 СМЕЩЕНИЙ
          ; Регистровая адресация
0009  B8 01F4          mov ax,n1
000C  8B C8          mov cx,ax
000E  B3 24          mov bl,E0L
0010  B7 CE          mov bh,n2

```

Microsoft (R) Macro Assembler Version 5.10

10/3/21

23:36:31

Page 1-2

```

          ; 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]

```

lb2.asm(51): error A2052: Improper operand type

```

; Базированная адресация
0020  8A 47 03          mov al,[bx]+3
0023  8B 4F 03          mov cx,3[bx]

; Индексная 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]
lb2.asm(60): 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]
lb2.asm(65): warning A4031: Operand types must match
003C  8B 85 0022 R      mov ax,matr[bx*4][di]
lb2.asm(66): error A2055: Illegal register value

; ПРОВЕРКА РЕЖИМОВ 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]
lb2.asm(90): error A2046: Multiple base registers
0061 3E: 8B 83 0016 R      mov ax,matr[bp+di+si]
lb2.asm(91): error A2047: Multiple index registers

```

```

; Использование сегмента стека
Microsoft (R) Macro Assembler Version 5.10      10/3/21
23:36:31
Page 1-3

```

```

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 CA 0002          ret 2
0076                Main ENDP
lb2.asm(99): error A2006: Phase error between passes
0076                CODE ENDS
                END Main

```

```

Microsoft (R) Macro Assembler Version 5.10      10/3/21
23:36:31
Symbols-1

```

#### Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK . . . . .	0018	PARA		STACK
CODE . . . . .	0076	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 =
0076			
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	lb2	
@VERSION . . . . .	TEXT	510	

101 Source Lines

101 Total Lines

19 Symbols

47824 + 459436 Bytes symbol space free

2 Warning Errors

5 Severe Errors

Название файла: LB2\_CORR.LST

Microsoft (R) Macro Assembler Version 5.10

10/5/21

18:13:01

Page 1-1

; Программа изучения режимов адресации процессора IntelX86

```
= 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 08 07 06 05 01 02  vec1 DB 8,7,6,5,1,2,3,4
          03 04
000E E2 D8 1E 28 F6 EC  vec2 DB -30,-40,30,40,-10,-20,10,20
          0A 14
0016 FF FE FD FC 08 07  matr   DB   -1,-2,-3,-4,8,7,6,5,-5,-6,-7,-
8,4,3,2,1
          06 05 FB FA F9 F8
          04 03 02 01
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 СМЕЩЕНИЙ
; Регистровая адресация
0009  B8 01F4          mov ax,n1
000C  8B C8          mov cx,ax
000E  B3 24          mov bl,E0L
0010  B7 CE          mov bh,n2

```

Microsoft (R) Macro Assembler Version 5.10

10/5/21

18:13:01

Page 1-2

```

; 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]

```

```

; Индексная адресация
0026 BF 0002          mov di,ind
0029 8A 85 000E R     mov al,vec2[di]
;    mov cx,vec2[di]

; Адресация с базированием и индексированием
002D BB 0003          mov bx,3
0030 8A 81 0016 R     mov al,matr[bx][di]
;    mov cx,matr[bx][di]
;    mov ax,matr[bx*4][di]

; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
; Переопределение сегмента

; ----- вариант 1
0034 B8 ---- R       mov ax, SEG vec2
0037 8E C0           mov es, ax
0039 26: 8B 07       mov ax, es:[bx]
003C B8 0000         mov ax, 0

; ----- вариант 2
003F 8E C0           mov es, ax
0041 1E             push ds
0042 07             pop es
0043 26: 8B 4F FF     mov cx, es:[bx-1]
0047 91             xchg cx,ax

; ----- вариант 3
0048 BF 0002          mov di,ind
004B 26: 89 01       mov es:[bx+di],ax

; ----- вариант 4
004E 8B EC           mov bp,sp
;    mov ax,matr[bp+bx]
;    mov ax,matr[bp+di+si]

```

; Использование сегмента стека

Microsoft (R) Macro Assembler Version 5.10

10/5/21

18:13:01

Page 1-3

```
0050 FF 36 0000 R      push mem1
0054 FF 36 0002 R      push mem2
0058 8B EC            mov bp,sp
005A 8B 56 02          mov dx,[bp]+2
005D CA 0002          ret 2
0060                      Main ENDP
0060                      CODE ENDS
                      END Main
```

Microsoft (R) Macro Assembler Version 5.10

10/5/21

18:13:01

Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK . . . . .	0018	PARA	STACK	
CODE . . . . .	0060	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 =
0060			

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	lb2
@VERSION . . . . .	TEXT	510

101 Source Lines

101 Total Lines

19 Symbols

47812 + 459448 Bytes symbol space free

0 Warning Errors

0 Severe Errors