

# **Лабараторная работа №08.**

## **НПИбд-03-24**

**Подготовил:**

Гелдиев Ыхлас. Студенческий номер: 1032249184

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
2.1	Реализация циклов в NASM . . . . .	6
2.2	Обработка аргументов командной строки . . . . .	12
<b>3</b>	<b>Задание для самостоятельной работы</b>	<b>18</b>
<b>4</b>	<b>Выводы</b>	<b>21</b>

# Список иллюстраций

2.1	Создал lab8-1.asm . . . . .	6
2.2	Заполнил lab8-1 . . . . .	7
2.3	Запуск lab8-1 . . . . .	8
2.4	Изменил текст lab8-1 . . . . .	9
2.5	Запуск измененного lab8-1 . . . . .	10
2.6	Использование стека в программе . . . . .	11
2.7	Проверка выполнения . . . . .	12
2.8	Создание lab8-2 . . . . .	12
2.9	Заполнение lab8-2 . . . . .	13
2.10	Запуск lab8-2 с аргументами и без . . . . .	13
2.11	Создание lab8-3.asm . . . . .	14
2.12	Заполнение lab8-3.asm . . . . .	15
2.13	Проверка lab8-3.asm . . . . .	15
2.14	Изменение программы для вычисления произведения . . . . .	16
2.15	Проверка корректности программы . . . . .	17
3.1	Программа main.asm . . . . .	19
3.2	Запуск main . . . . .	20

## **Список таблиц**

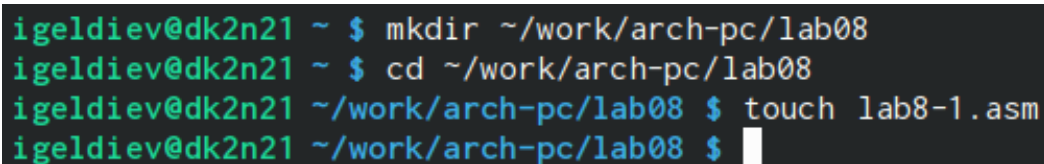
# 1 Цель работы

Освоить написание программ с использованием циклов и обработкой аргументов командной строки.

## 2 Выполнение лабораторной работы

### 2.1 Реализация циклов в NASM

1. Создал каталог для программ, перешел в него и создал файл lab8-1.asm (рис. 2.1)

A terminal window with a dark background and light green text. It shows four lines of commands and their execution. The first line creates a directory, the second changes to it, the third creates a file, and the fourth shows the prompt. The user's name is igeldiev and the host is dk2n21.

```
igeldiev@dk2n21 ~ $ mkdir ~/work/arch-pc/lab08
igeldiev@dk2n21 ~ $ cd ~/work/arch-pc/lab08
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ touch lab8-1.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 2.1: Создал lab8-1.asm

2. Заполнил lab8-1.asm (рис. 2.2)

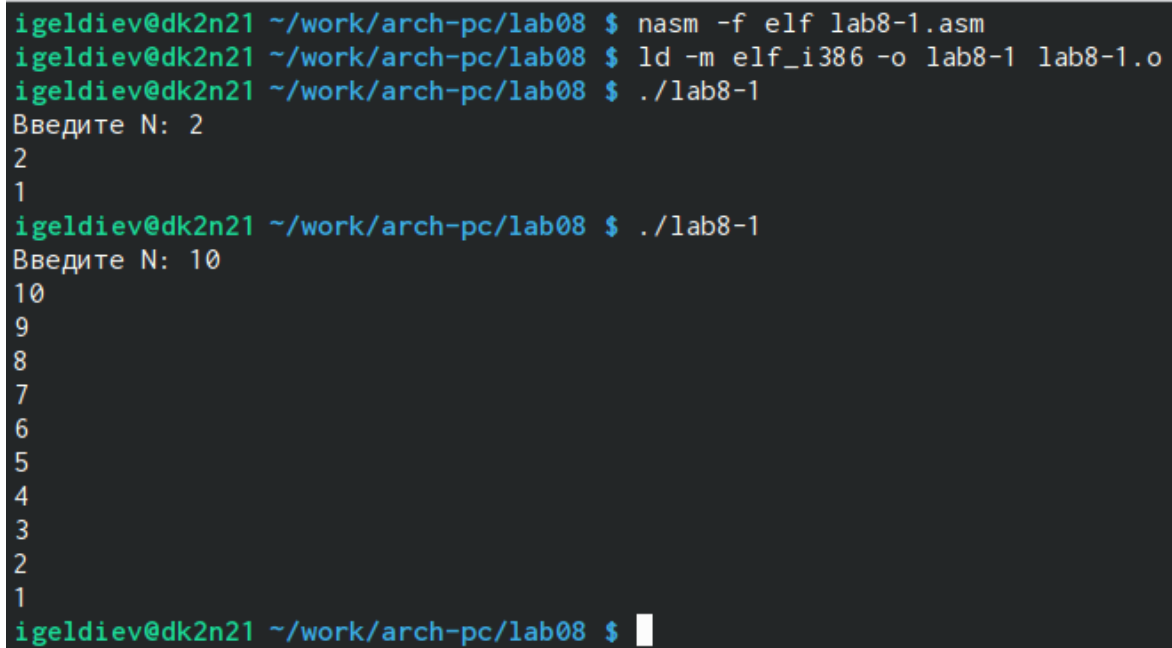
```

1 %include 'in_out.asm'
2
3 SECTION .data
4     msg1 db 'Введите N: ',0h
5
6 SECTION .bss
7     N: resb 10
8
9 SECTION .text
10    global _start
11
12 _start:
13
14 ; ----- Вывод сообщения 'Введите N: '
15     mov eax,msg1
16     call sprint
17
18 ; ----- Ввод 'N'
19
20     mov ecx, N
21     mov edx, 10
22     call sread
23
24 ; ----- Преобразование 'N' из символа в число
25
26     mov eax,N
27     call atoi
28     mov [N], eax
29
30 ; ----- Организация цикла
31     mov ecx,[N]      ; Счетчик цикла, 'ecx=N'
32 label:
33     mov [N],ecx
34     mov eax,[N]
35     call iprintLF    ; Вывод значения 'N'
36     loop label       ; 'ecx=ecx-1' и если 'ecx' не '0'
37                     ; переход на 'label'
38     call quit
39

```

Рис. 2.2: Заполнил lab8-1

3. Создал исполняемый файл и запустил его (рис. 2.3)



```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 2
2
1
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 2.3: Запуск lab8-1

4. Изменил текст программы (рис. 2.4)



```

1 %include 'in_out.asm'
2
3 SECTION .data
4   msg1 db 'Введите N: ',0h
5
6 SECTION .bss
7   N: resb 10
8
9 SECTION .text
10  global _start
11
12 _start:
13
14 ; ----- Вывод сообщения 'Введите N: '
15   mov eax,msg1
16   call sprint
17
18 ; ----- Ввод 'N'
19
20   mov ecx, N
21   mov edx, 10
22   call sread
23
24 ; ----- Преобразование 'N' из символа в число
25
26   mov eax,N
27   call atoi
28   mov [N], eax
29
30 ; ----- Организация цикла
31   mov ecx,[N]      ; Счетчик цикла, 'ecx=N'
32 label:
33   sub ecx,1        ; 'ecx=ecx-1'
34   mov [N],ecx
35   mov eax,[N]
36   call iprintLF    ; Вывод значения 'N'
37   loop label       ; 'ecx=ecx-1' и если 'ecx' не '0'
38                   ; переход на 'label'
39   call quit
40

```

Рис. 2.4: Изменил текст lab8-1

5. Создал исполняемый файл и запустил его (рис. 2.5)

```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
7
5
3
1
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 2.5: Запуск измененного lab8-1

6. Изменил текст программы, с использованием стека в программу для сохранения корректности работы программы (рис. 2.6)

```

1 %include 'in_out.asm'
2
3 SECTION .data
4 msg1 db 'Введите N: ',0h
5
6 SECTION .bss
7 N: resb 10
8
9 SECTION .text
10 global _start
11
12 _start:
13
14 ; ----- Вывод сообщения 'Введите N: '
15 mov eax,msg1
16 call sprint
17
18 ; ----- Ввод 'N'
19
20 mov ecx, N
21 mov edx, 10
22 call sread
23
24 ; ----- Преобразование 'N' из символа в число
25
26 mov eax,N
27 call atoi
28 mov [N], eax
29
30 ; ----- Организация цикла
31 mov ecx,[N] ; Счетчик цикла, 'ecx=N'
32 label:
33 push ecx
34 sub ecx,1 ; 'ecx=ecx-1'
35 mov [N],ecx
36 mov eax,[N]
37 call iprintLF ; Вывод значения 'N'
38 pop ecx
39 loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
40 ; переход на 'label'
41 call quit
42

```

Рис. 2.6: Использование стека в программе

7. Проверка выполнения (рис. 2.7)

```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 2.7: Проверка выполнения

## 2.2 Обработка аргументов командной строки

8. Создал файл lab8-2.asm (рис. 2.8)

```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ touch lab8-2.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 2.8: Создание lab8-2

9. Заполнил lab8-2.asm (рис. 2.9)

```

1 ;-----
2 ; Обработка аргументов командной строки
3 ;-----
4 %include 'in_out.asm'
5
6 SECTION .text
7 global _start
8
9 _start:
10  pop ecx          ; Извлекаем из стека в 'ecx' количество
11                   ; аргументов (первое значение в стеке)
12  pop edx          ; Извлекаем из стека в 'edx' имя программы
13                   ; (второе значение в стеке)
14  sub ecx,1        ; Уменьшаем 'ecx' на 1 (количество
15                   ; аргументов без названия программы)
16
17 _next:
18  cmp ecx, 0       ; проверяем, есть ли еще аргументы
19  jz _end          ; если аргументов нет выходим из цикла
20                   ; (переход на метку '_end')
21  pop eax          ; иначе извлекаем аргумент из стека
22  call sprintf      ; вызываем функцию печати
23  loop _next       ; переход к обработке следующего
24                   ; аргумента (переход на метку 'next')
25 _end:
26  call quit
27

```

Рис. 2.9: Заполнение lab8-2

10. Создал исполняемый файл и запустил его, указав аргументы. Программа обработала 4 аргумента (рис. 2.10)

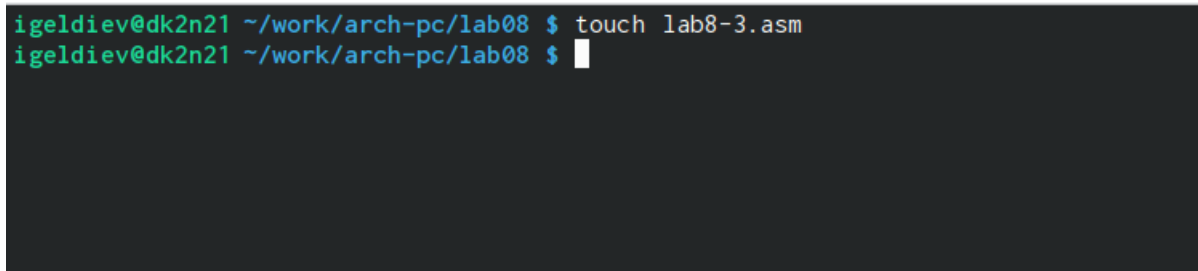
```

igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
igeldiev@dk2n21 ~/work/arch-pc/lab08 $

```

Рис. 2.10: Запуск lab8-2 с аргументами и без

11. Создал файл lab8-3.asm (рис. [ @-fig:011])

A terminal window with a dark background. The prompt is 'igeldiev@dk2n21' and the current directory is '~/work/arch-pc/lab08'. The user has entered the command 'touch lab8-3.asm' and pressed enter. The prompt is now on a new line.

```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ touch lab8-3.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 2.11: Создание lab8-3.asm

12. Заполнил lab8-3.asm (рис. 2.12)

```

1 %include 'in_out.asm'
2
3 SECTION .data
4 msg db "Результат: ",0
5
6 SECTION .text
7 global _start
8
9 _start:
10 pop ecx      ; Извлекаем из стека в 'ecx' количество
11              ; аргументов (первое значение в стеке)
12 pop edx      ; Извлекаем из стека в 'edx' имя программы
13              ; (второе значение в стеке)
14 sub ecx,1     ; Уменьшаем 'ecx' на 1 (количество
15              ; аргументов без названия программы)
16 mov esi,0     ; Используем 'esi' для хранения
17              ; промежуточных сумм
18 next:
19 cmp ecx,0h    ; проверяем, есть ли еще аргументы
20 jz _end       ; если аргументов нет выходим из цикла
21              ; (переход на метку '_end')
22
23 pop eax       ; иначе извлекаем следующий аргумент из стека
24 call atoi     ; преобразуем символ в число
25 add esi,eax   ; добавляем к промежуточной сумме
26              ; след. аргумент 'esi=esi+eax'
27
28 loop next     ; переход к обработке следующего аргумента
29
30 _end:
31 mov eax,msg   ; вывод сообщения "Результат: "
32 call sprint
33 mov eax,esi   ; записываем сумму в регистр 'eax'
34 call iprintLF ; печать результата
35
36 call quit     ; завершение программы

```

Рис. 2.12: Заполнение lab8-3.asm

13. Создал исполняемый файл и запустил его (рис. 2.13)

```

igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 47
igeldiev@dk2n21 ~/work/arch-pc/lab08 $

```

Рис. 2.13: Проверка lab8-3.asm

14. Изменил текст программы для вычисления произведения аргументов командной строки (рис. 2.14)

```
1 %include 'in_out.asm'
2
3 SECTION .data
4 msg db "Результат: ",0
5
6 SECTION .text
7 global _start
8
9 _start:
10 pop ecx      ; Извлекаем из стека в 'ecx' количество
11              ; аргументов (первое значение в стеке)
12 pop edx      ; Извлекаем из стека в 'edx' имя программы
13              ; (второе значение в стеке)
14 sub ecx,1    ; Уменьшаем 'ecx' на 1 (количество
15              ; аргументов без названия программы)
16 mov esi,1    ; Используем 'esi' для хранения
17              ; промежуточных сумм
18 next:
19 cmp ecx,0h   ; проверяем, есть ли еще аргументы
20 jz _end      ; если аргументов нет выходим из цикла
21              ; (переход на метку '_end')
22
23 pop eax      ; иначе извлекаем следующий аргумент из стека
24 call atoi    ; преобразуем символ в число
25
26 ; ----- added by user -----
27 ; add esi,eax ; добавляем к промежуточной сумме
28              ; след. аргумент 'esi=esi+eax'
29
30 mul esi      ; eax = eax * esi
31 mov esi, eax ; esi = eax
32
33 ; -----
34
35 loop next    ; переход к обработке следующего аргумента
36
37 _end:
38 mov eax,msg  ; вывод сообщения "Результат: "
39 call sprint
40 mov eax, esi ; записываем сумму в регистр 'eax'
41 call iprintLF ; печать результата
42
43 call quit    ; завершение программы
```

Рис. 2.14: Изменение программы для вычисления произведения

15. Проверка программы (рис. 2.15)



```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ gedit lab8-3.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 2 2
Результат: 0
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ gedit lab8-3.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 2 2
Результат: 4
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 2 2 3
Результат: 12
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 12 2
Результат: 24
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./lab8-3 30 12 32 0
Результат: 0
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ █
```

Рис. 2.15: Проверка корректности программы

### **3 Задание для самостоятельной работы**

1. Программа для нахождения суммы значений функции  $f(x)$  для разных  $x$   
(рис. 3.1) (рис. 3.2)

```

1 ;-----
2 ; f(x) = 4x+2, var 5
3 ;-----
4
5
6 %include 'in_out.asm'
7
8 SECTION .data
9     msg1 db 'Функция: f(x)=4x+2',0
10    msg2 db 'Результат: ',0
11
12 SECTION .text
13 global _start
14
15 _start:
16     mov eax, msg1
17     call sprintf
18
19     pop ecx
20     pop edx
21     dec ecx
22     mov esi,0
23
24 next:
25     cmp ecx,0h
26     jz _end
27
28     pop eax
29     call atoi
30
31     mov edx, 4
32     mul edx
33     add eax, 2
34
35     add esi, eax
36
37     loop next
38
39 _end:
40     mov eax,msg2
41     call sprintf
42
43     mov eax,esi
44     call iprintLF
45
46     call quit
47

```

Рис. 3.1: Программа main.asm

```
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf main.asm
main.asm:31: error: invalid combination of opcode and operands
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ nasm -f elf main.asm
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o main main.o
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./main
Функция:  $f(x)=4x+2$ 
Результат: 0
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./main 1
Функция:  $f(x)=4x+2$ 
Результат: 6
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./main 1 2
Функция:  $f(x)=4x+2$ 
Результат: 16
igeldiev@dk2n21 ~/work/arch-pc/lab08 $ ./main 1 2 3
Функция:  $f(x)=4x+2$ 
Результат: 30
igeldiev@dk2n21 ~/work/arch-pc/lab08 $
```

Рис. 3.2: Запуск main

## 4 Выводы

Я научился писать программы с использованием циклов и обработкой аргументов командной строки.