

Лабораторная работа №8

Программирование цикла. Обработка аргументов командной строки

Жукова Арина Александровна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Реализация циклов в NASM	6
2.2	Обработка аргументов командной строки	8
3	Задания для самостоятельной работы	11
4	Выводы	13
	Список литературы	14

Список иллюстраций

2.1	Введение текста программы	6
2.2	Проверка работы программы	7
2.3	Изменение строк программы	7
2.4	Изменение строк программы	7
2.5	Проверка работы программы	8
2.6	Введение текста программы в файл	8
2.7	Проверка работы программы с аргументами	9
2.8	Введение текста программы в файл	9
2.9	Проверка работы программы с аргументами	9
2.10	Изменение текста программы	10
2.11	Запуск программы с изменениями	10
3.1	Запуск программы	12

Список таблиц

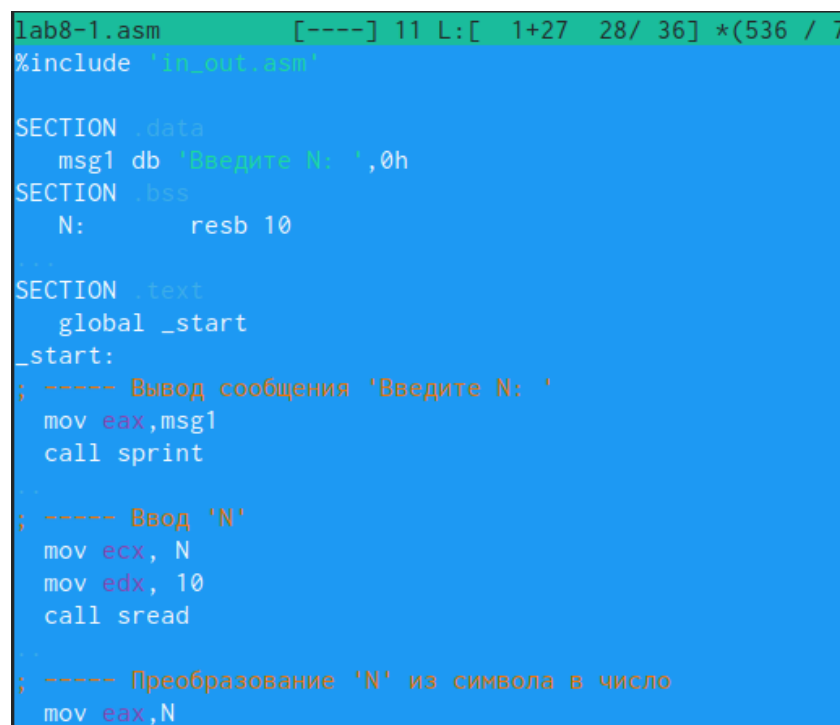
1 Цель работы

Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

2 Выполнение лабораторной работы

2.1 Реализация циклов в NASM

Создаём каталог для программ лабораторной работы № 8, перейдём в него и создаём файл lab8-1.asm. Введём в него текст программы из листинга 8.1 (рис. 2.1).



```
lab8-1.asm [----] 11 L: [ 1+27 28/ 36] *(536 / 7
%include 'in_out.asm'

SECTION .data
    msg1 db 'Введите N: ',0h
SECTION .bss
    N:      resb 10
...
SECTION .text
    global _start
_start:
; ----- Вывод сообщения 'Введите N: '
    mov eax,msg1
    call sprint
; ----- Ввод 'N'
    mov ecx, N
    mov edx, 10
    call sread
; ----- Преобразование 'N' из символа в число
    mov eax,N
```

Рис. 2.1: Введение текста программы

Создаём исполняемый файл и проверяем его работу (рис. 2.2).

```

aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
5
4
3
2
1

```

Рис. 2.2: Проверка работы программы

Внесем изменения в некоторые строки текста программы (рис. 2.3).

```

; ----- Организация цикла
mov ecx,[N]          ; Счетчик цикла, 'ecx=N'
label:
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label          ; 'ecx=ecx-1' и если 'ecx' не '0'

```

Рис. 2.3: Изменение строк программы

Число проходов цикла не соответствует значению N, введенного с клавиатуры.

Внесём изменение в строки программы, введя команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла (рис. 2.4).

```

label:
push ecx
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
pop ecx
loop label          ; 'ecx=ecx-1' и если 'ecx' не '0'

```

Рис. 2.4: Изменение строк программы

Создаём исполняемый файл и проверяем его работу (рис. 2.5).

```

aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
4
3
2
1
0

```

Рис. 2.5: Проверка работы программы

В данном случае число проходов цикла совпадает со значением N, введёному с клавиатуры.

2.2 Обработка аргументов командной строки

Создадим файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и введём в него текст программы из листинга 8.2 (рис. 2.6).

```

lab8-2.asm [----] 0 L:[ 1+22
#include 'in_out.asm'

SECTION .text
global _start

_start:
    pop ecx

    pop edx

    sub ecx, 1
    ...
next:
    cmp ecx, 0

```

Рис. 2.6: Введение текста программы в файл

Создадим исполняемый файл и запустим его, указав аргументы: аргумент1 аргумент 2 'аргумент 3 (рис. 2.7).

```
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
```

Рис. 2.7: Проверка работы программы с аргументами

Программой было обработано 4 аргумента.

Создадим файл lab8-3.asm в каталоге ~/work/arch-pc/lab08 и введём в него текст программы из листинга 8.3 (рис. 2.8).

```
lab8-3.asm [----] 15 L: [ 1+24 25/ 30] *(278 /
#include "in_out.asm"

SECTION .data
msg db "Результат: ",0

SECTION .text
global _start

_start:
    pop ecx
    pop edx
    sub ecx,1
    mov esi,0
```

Рис. 2.8: Введение текста программы в файл

Создадим исполняемый файл и проверим его работу, указав аргумент (рис. 2.9).

```
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 47
```

Рис. 2.9: Проверка работы программы с аргументами

Изменим текст программы из листинга 8.3 для вычисления произведения аргументов командной строки (рис. 2.10).

```
next:
    cmp ecx, 0h
    jz _end
    pop eax
    call atoi
    mov ebx, eax
    mov eax, esi
    mul ebx
    mov esi, eax

    loop next
```

Рис. 2.10: Изменение текста программы

Проверим работу программы (рис. 2.11).

```
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-3 8 7 6
Результат: 336
```

Рис. 2.11: Запуск программы с изменениями

Для корректной работы программы изменим значение `esi` на 1, в `ebx` вписываем значение аргумента, в `eax` значение того на что умножается, перемножаем `ebx` и `eax`, записываем полученное в `esi`.

3 Задания для самостоятельной работы

Программа, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$, т.е. программа должна выводить значение $f(x_1) + f(x_2) + \dots + f(x_n)$. Значения x_i передаются как аргументы. $f(x) = 2x + 15$

```
%include      'in_out.asm'
```

```
SECTION .data
```

```
msg db "Результат: ",0
```

```
SECTION .text
```

```
global _start
```

```
_start:
```

```
    pop ecx
```

```
    pop edx
```

```
    sub ecx, 1
```

```
    mov esi, 0
```

```
next:
```

```
    cmp ecx, 0h
```

```
    jz _end
```

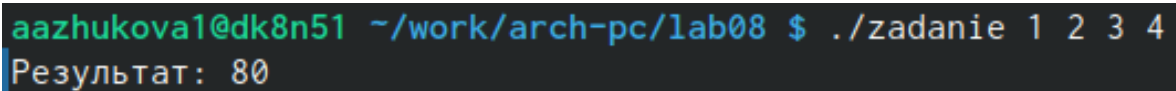
```
    pop eax
```

```
call atoi
mov ebx,2
mul ebx
add eax,15
add esi,eax
loop next
```

_end:

```
mov eax,msg
call sprint
mov eax,esi
call iprintLF
call quit
```

Создадим исполняемый файл и проверим его работу (рис. 3.1).

A terminal window with a dark background. The prompt is 'aazhukova1@dk8n51 ~/work/arch-pc/lab08 \$'. The command './zadanie 1 2 3 4' has been entered. The output 'Результат: 80' is displayed on the next line.

```
aazhukova1@dk8n51 ~/work/arch-pc/lab08 $ ./zadanie 1 2 3 4
Результат: 80
```

Рис. 3.1: Запуск программы

4 Выводы

В ходе выполнения лабораторной работы мы приобрели навыки написания программ с использованием циклов и обработкой аргументов командной строки.

Список литературы