

Лабораторная работа №8

Программирование цикла. Обработка аргументов командной строки.

Краснопер Данила Олегович

Содержание

1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Самостотельная работа	12
	Вывод	14

1 Цель работы

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

2 Выполнение лабораторной работы

- 1) Я создал каталог lab8 и файл lab8-1.asm

```
dokrasnoper@dk3n51 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/labs/lab07/report $ mkdir ~/work/arch-pc/lab08
dokrasnoper@dk3n51 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/labs/lab07/report $ cd ~/work/arch-pc/lab08
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ touch lab8-1.asm
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $
```

Рис. 2.1: Создание файла и каталога

- 2) В файл я ввел текст первой программы и создал исполняемый файл.

```

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

```

Рис. 2.2: Текст программы

```

dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
5
4
3
2
1
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ 

```

Рис. 2.3: Запуск программы и проверка результата

3) Я изменил текст программы, в теле цикла label добавил строку `sub eax,1`. Циклы закольцевался и стал бесконечным.

```
mov ecx,[N] ; счетчик цикла  
label:  
sub ecx,1 ; 'ecx=ecx-1'  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
loop label  
□
```

Рис. 2.4: Измененный текст программы

4294578192
4294578190
4294578188
4294578186
4294578184
4294578182
4294578180
4294578178
4294578176
4294578174
4294578172
4294578170
4294578168
4294578166
4294578164
4294578162
4294578160
4294578158

Рис. 2.5: Запуск программы

4)Я изменил текст программы так, чтобы цикл и счетчик работал правильно. По итогу после изменения программы, яисло проходки циклов стал соответствовать числу введенному с клавиатуры.

```
mov ecx,[N] ; Счетчик цикла, `ecx=N`  
label:  
push ecx ; добавление значения ecx в стек  
sub ecx,1  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
pop ecx ; извлечение значения ecx из стека  
loop label
```

Рис. 2.6: Редактирование текста программы

```
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-1  
Введите N: 5  
4  
3  
2  
1  
0
```

Рис. 2.7: Запуск измененной программы

5)Я создал файл lab8-2.asm и ввел туда программу, которая выводит все аргументы, которые ввели. Программа выводит все 3 аргумента которые ввели, но в разной вариации.


```

lab8-2.asm      [-----]  9 L:[  1+19  20/ 20] *(943
#include 'in_out.asm'
SECTION .text
global _start
_start:
pop ecx ; Извлекаем из стека в 'ecx' количество
; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в 'edx' имя программы
; (второе значение в стеке)
sub ecx, 1 ; Уменьшаем 'ecx' на 1 (количество
; аргументов без названия программы)
next:
cmp ecx, 0 ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку '_end')
pop eax ; иначе извлекаем аргумент из стека
call sprintf ; вызываем функцию печати
loop next ; переход к обработке следующего
; аргумента (переход на метку 'next')
_end:
call quit

```

Рис. 2.8: Текст программы для вывода аргументов

```

dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $

```

Рис. 2.9: Результаты работы программы

- 6) Я создал файл lab8-3.asm. Ввел текст программы и запустил ее. Программа вывела сумму чисел, которые я ввел.

```

; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в 'edx' имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем 'ecx' на 1 (количество
; аргументов без названия программы)
mov esi, 0 ; Используем 'esi' для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку '_end')
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
add esi,eax ; добавляем к промежуточной сумме
; след. аргумент 'esi=esi+eax'
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint
mov eax, esi ; записываем сумму в регистр 'eax'
call iprintLF ; печать результата
call quit ; завершение программы

```

Рис. 2.10: Текст программы lab9-3

```

dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-3
Результат: 0
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-3 13 4 85
Результат: 102

```

Рис. 2.11: Результат работы программы

- 7) Я изменил программу, чтобы она выводила произведение введенных чисел.

```

pop edx ; Извлекаем из стека в 'edx' имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем 'ecx' на 1 (количество
; аргументов без названия программы)
mov esi,1 ; Используем 'esi' для хранения
; промежуточных сумм
mov eax,1
next:
cmp ecx,0 ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку '_end')
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mov ebx,eax
mov ecx,esi
mul ebx
mov esi,eax
loop next

_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint

```

Рис. 2.12: Текст программы с произведением чисел

```

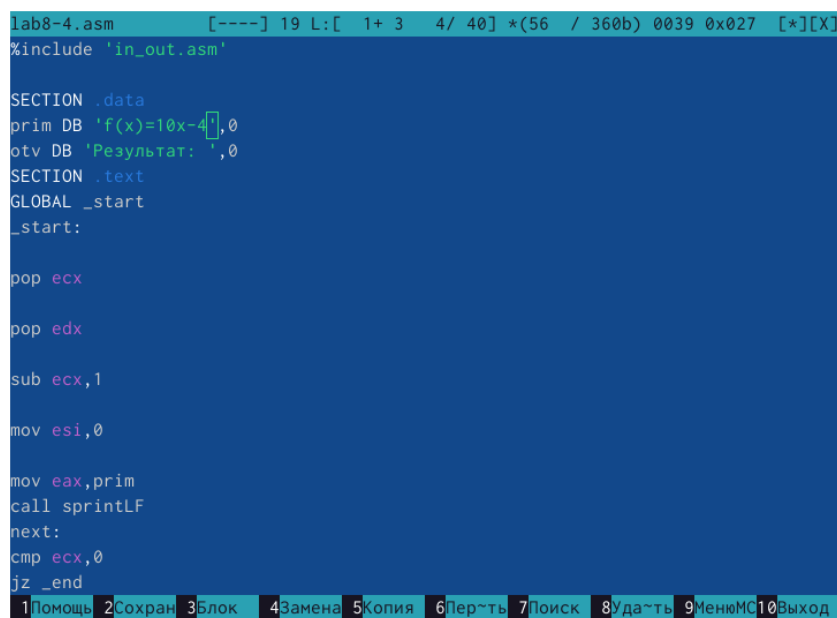
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-3 3 1 5 2
Результат: 30
dokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-3 2 5 2 0
Результат: 0

```

Рис. 2.13: Результаты работы программы с произведением

3 Самостоятельная работа

Я написал программу, которая выводит сумму всех решений примера. В лабораторной работе №6, я получил 9 вариантов, поэтому я писал программу для девятого варианта. Введенные числа я придумал сам, и посчитал их, чтобы проверить работу программы.



```
lab8-4.asm      [----] 19 L:[ 1+ 3 4/ 40] *(56 / 360b) 0039 0x027 [*][X]
#include 'in_out.asm'

SECTION .data
prim DB 'f(x)=10x-4',0
otv DB 'Результат: ',0
SECTION .text
GLOBAL _start
_start:

pop ecx

pop edx

sub ecx,1

mov esi,0

mov eax,prim
call sprintLF
next:
cmp ecx,0
jz _end

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюМС10Выход
```

Рис. 3.1: Текст программы в самостоятельной работе

```
lokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4
f(x)=10x-4
Результат: 84
lokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4 8
f(x)=10x-4
Результат: 160
lokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ ./lab8-4 3 2 7 0
f(x)=10x-4
Результат: 104
lokrasnoper@dk3n51 ~/work/arch-pc/lab08 $ █
```

Рис. 3.2: Результаты работы программы

Вывод

Я приобрел навыки написания программы с использованием цикла.