

Лабораторная работа №8

Программирование цикла. Обработка аргументов командной строки.

Шершунов Демьян

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Самостотельная работа	14
	Вывод	16

Список иллюстраций

2.1	Создание файла и каталога	6
2.2	Текст программы	7
2.3	Запуск программы и проверка результата	7
2.4	Измененный текст программы	8
2.5	Запуск программы	9
2.6	Редактирование текста программы	10
2.7	Запуск измененной программы	10
2.8	Текст программы для вывода аргументов	11
2.9	Результаты работы программы	11
2.10	Текст программы lab8-3	12
2.11	Результат работы программы	12
2.12	Текст программы с произведением чисел	13
2.13	Результаты работы программы с произведением	13
3.1	Текст программы в самостоятельной работе	14
3.2	Результаты работы программы	15

Список таблиц

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

```
demyanshershunov@vbox:~$ mkdir ~/work/arch-pc/lab08  
demyanshershunov@vbox:~$ cd ~/work/arch-pc/lab08  
demyanshershunov@vbox:~/work/arch-pc/lab08$ touch lab8-1.asm  
demyanshershunov@vbox:~/work/arch-pc/lab08$
```

Рис. 2.1: Создание файла и каталога

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

```
Открыть ▾ + lab8-1.asm
~/work/arch-pc/lab08

;-----
; Программа вывода значений регистра 'eax'
;-----


%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 printf
; ---- Ввод 'N'
mov ecx, N
mov edx, 10
call read
; ---- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ---- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'
label:
mov [N],ecx
mov eax,[N]
call printHE ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call exit
```

Рис. 2.2: Текст программы

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

Рис. 2.3: Запуск программы и проверка результата

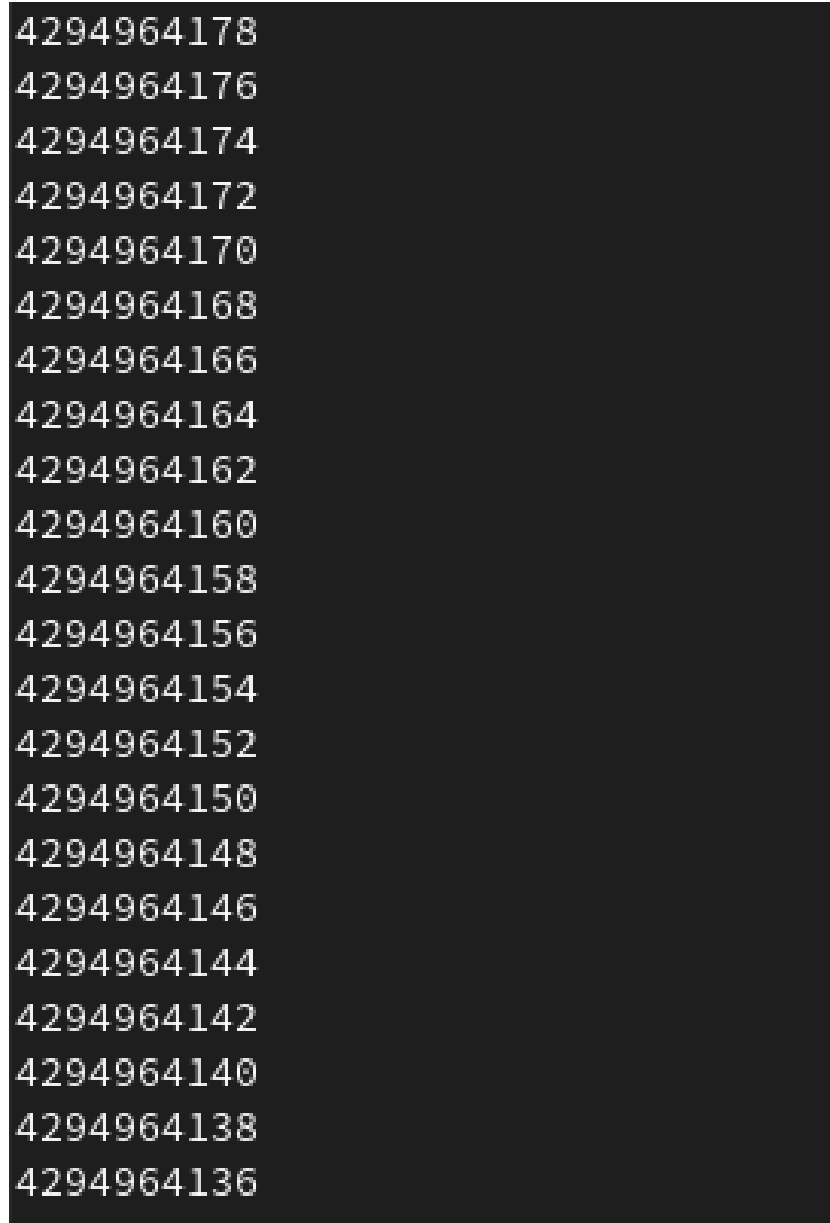
3)Я изменил текст программы, в теле цикла label добавил строку `sub eax,1`. Циклы закольцевался и стал бесконечным.

Открыть ▾ 

lab8-1.asm
~/work/arch-pc/lab08

```
;-----  
; Программа вывода значений регистра 'ecx'  
;-----  
  
%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  
call atoi  
mov [N],eax  
; ----- Организация цикла  
mov ecx,[N] ; Счетчик цикла, `ecx=N`  
label:  
sub ecx,1 ; `ecx=ecx-1`  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
loop label  
call quit
```


Рис. 2.4: Измененный текст программы



```
4294964178
4294964176
4294964174
4294964172
4294964170
4294964168
4294964166
4294964164
4294964162
4294964160
4294964158
4294964156
4294964154
4294964152
4294964150
4294964148
4294964146
4294964144
4294964142
4294964140
4294964138
4294964136
4294964134
```

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.

Открыть ▾ 

• lab8-1.asm
~/work/arch-pc/lab08

```
;-----  
; Программа вывода значений регистра 'ecx'  
;-----  
%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  
call atoi  
mov [N],eax  
; ----- Организация цикла  
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  
call quit
```

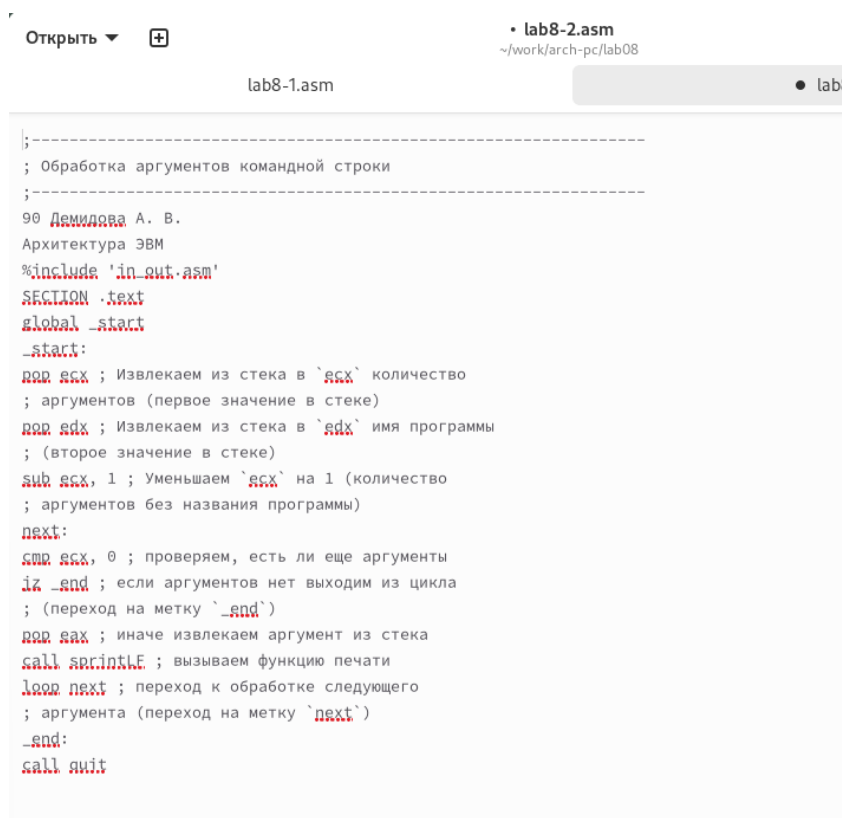
Рис. 2.6: Редактирование текста программы

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

Рис. 2.7: Запуск измененной программы

5)Я создал файл lab8-2.asm и ввел туда программу, которая выводит все аргу-

мент, которые ввели. Программа выводит все 3 аргумента которые ввели, но в разной вариации.

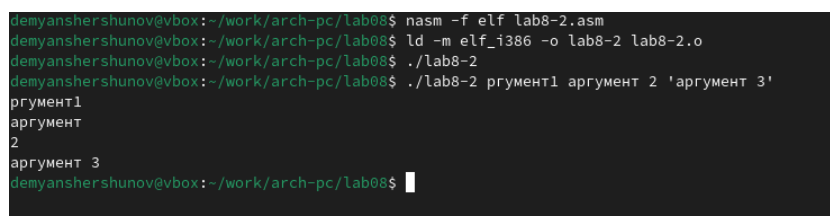


```

;-----
; Обработка аргументов командной строки
;-----
90 Демидова А. В.
Архитектура ЭВМ
%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 printf ; 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



```

demyanshershunov@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
demyanshershunov@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
demyanshershunov@vbox:~/work/arch-pc/lab08$ ./lab8-2
аргумент1
аргумент
2
аргумент 3
demyanshershunov@vbox:~/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.

```
Открыть ▾ + lab8-3.asm
~/work/arch-pc/lab08
lab8-1.asm lab8-2.asm


%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
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)
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 lab8-3

```
demyanshershunov@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
demyanshershunov@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
demyanshershunov@vbox:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4
Результат: 10
demyanshershunov@vbox:~/work/arch-pc/lab08$
```

Рис. 2.11: Результат работы программы

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

Открыть  lab8-3.asm
~/work/arch-pc/lab08

lab8-1.asm | lab8-2.asm

```
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
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)
mov esi, 1
mov eax, 1
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
mov ebx,eax
mov eax,esi
mul ebx
mov 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. 2.12: Текст программы с произведением чисел

```
demyanshershunov@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
demyanshershunov@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
demyanshershunov@vbox:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4
Результат: 24
demyanshershunov@vbox:~/work/arch-pc/lab08$
```

Рис. 2.13: Результаты работы программы с произведением

3 Самостоятельная работа

Я написал программу, которая выводит сумму всех решений примера. Введенные числа я придумал сам, и посчитал их, чтобы проверить работу программы.



```
lab8-4.asm
~/work/arch-pc/lab08

lab8-1.asm | lab8-2.asm | lab8-3.asm

%include 'in_out.asm'

SECTION .data
prim DB 'f(x)=15x+2',0
otv DB 'Результат: ',0
SECTION .text
GLOBAL _start
_start:

pop ecx

pop edx

sub ecx,1

mov esi,0

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

mov ebx,15
pop eax
call atoi
mul ebx

add eax,2

add esi,eax

loop next
```

Рис. 3.1: Текст программы в самостоятельной работе

```
demyanshershunov@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
demyanshershunov@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
demyanshershunov@vbox:~/work/arch-pc/lab08$ ./lab8-4 1 2 3 4
f(x)=15x+2
Результат: 158
demyanshershunov@vbox:~/work/arch-pc/lab08$
```

Рис. 3.2: Результаты работы программы

Вывод

Я приобрел навыки написания программы с использованием цикла.