

Отчёт по лабораторной работе №8

Архитектура компьютера НММбд-03-24

Туева Анастасия Юрьевна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выполнение самостоятельной работы	16
4	Выводы	19

Список иллюстраций

2.1	Создание файла	6
2.2	Редактирование файла	7
2.3	Запуск исполняемого файла	8
2.4	Редактирование файла	8
2.5	Запуск исполняемого файла	9
2.6	Редактирование файла	10
2.7	Запуск исполняемого файла	11
2.8	Редактирование файла	11
2.9	Запуск исполняемого файла	12
2.10	Файл листинга “lab8-3.asm”	12
2.11	Запуск исполняемого файла	13
2.12	Редактирование файла	14
2.13	Запуск исполняемого файла	15
3.1	Редактирование файла	17
3.2	Запуск исполняемого файла	18

Список таблиц

1 Цель работы

Приобрести навыки написания программ с использованием циклов и обработкой аргументов командной строки.

2 Выполнение лабораторной работы

Создаём каталог для программ лабораторной работы № 8, переходим в него и создаём файл “lab8-1.asm” (рис. 2.1).

```
aytueva@dk6n50 ~ $ mkdir ~/work/arch-pc/lab08  
aytueva@dk6n50 ~ $ cd ~/work/arch-pc/lab08  
aytueva@dk6n50 ~/work/arch-pc/lab08 $ touch lab8-1.asm  
aytueva@dk6n50 ~/work/arch-pc/lab08 $
```

Рис. 2.1: Создание файла

Открываем созданный файл “lab8-1.asm”, вставляем в него данный текст в соответствии с листингом 8.1. (рис. 2.2).

```

lab8-1.asm          [-M--] 21 L:[ 1+
#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 символ
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'

```

Рис. 2.2: Редактирование файла

Создаем исполняемый файл программы и запускаем его (рис. 2.3).

```

aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 5
5
4
3
2
1

```

Рис. 2.3: Запуск исполняемого файла

Изменяем в текст программы добавив изменение значение регистра “ecx” в цикле (рис. 2.4).

```

lab8-1.asm      [-M--]  9 L: [  8+21  29/
_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 iprintf
loop label
; переход на 'label'
call quit

```

Рис. 2.4: Редактирование файла

Создаем новый исполняемый файл программы и запускаем его (рис. 2.5).

```
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 4
3
1
```

Рис. 2.5: Запуск исполняемого файла

Изменяем в текст программы добавив команды “push” и “pop” для сохранения значения счетчика цикла “loop” (рис. 2.6).

```

lab8-1.asm          [-M--]  9 L:[ 10+20
mov eax,msg1
call sprint
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа
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
; переход на 'label'
call quit

```

Рис. 2.6: Редактирование файла

Создаем новый исполняемый файл программы и запускаем его (рис. 2.7).

```

aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 4
3
2
1
0

```

Рис. 2.7: Запуск исполняемого файла

Создаем новый файл “lab8-2.asm” с помощью команды “touch”.

Вводим в файл текст другой программы в соответствии с листингом 8.2. (рис. 2.8).

```

lab8-2.asm [-M--] 9 L:[ 1+19 20/
#include 'in_out.asm'
SECTION text
global _start
_start:
pop ecx ; Извлечен из стека в 'ecx' количе
; аргументов (первое значение в стеке)
pop edx ; Извлечен из стека в 'edx' ине пр
; (второе значение в стеке)
sub ecx, 1 ; Уменьшен 'ecx' на 1 (количест
; аргументов без названия программы)
next:
cmp ecx, 0 ; проверяем, есть ли еще аргумен
jz _end ; если аргументов нет выходим из ци
; (переход на метку '_end')
pop eax ; иначе извлечен аргумент из стека
call sprintf ; вызван функция печати
loop next ; переход к обработке следующего
; аргумента (переход на метку 'next')
_end:
call quit

```

Рис. 2.8: Редактирование файла

Создаем исполняемый файл программы и запускаем его, указав аргументы :

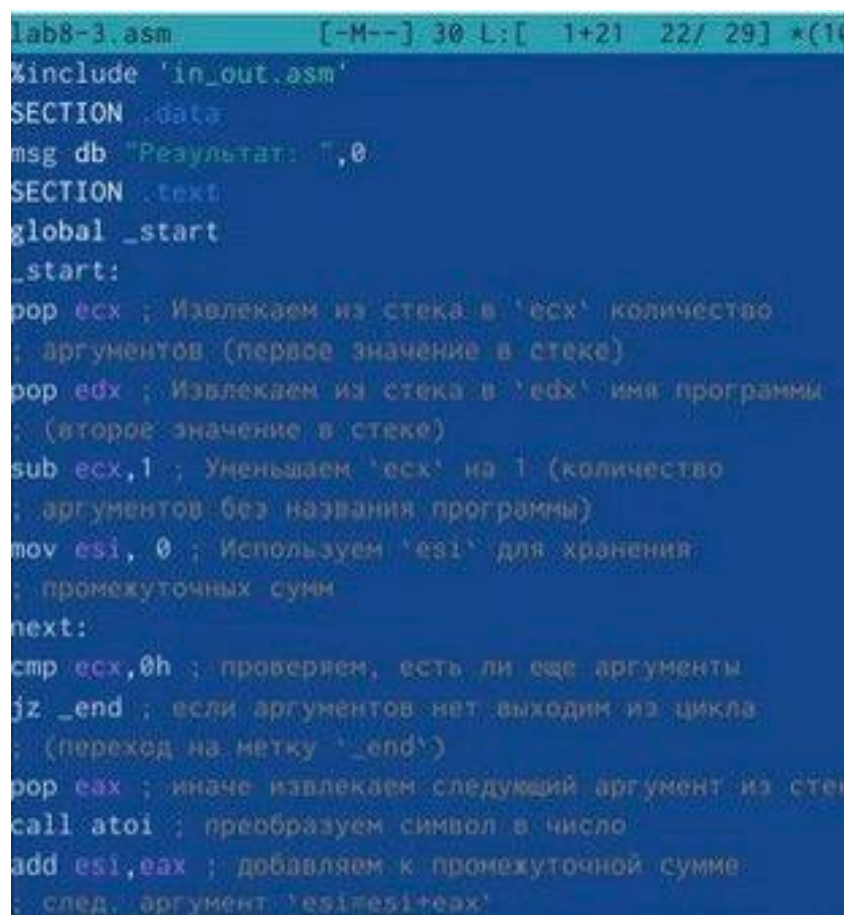
“аргумент1 аргумент 2 ‘аргумент 3’” (рис. 2.9).

```
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
```

Рис. 2.9: Запуск исполняемого файла

Создаем новый файл “lab8-3.asm” с помощью команды “touch”.

Вводим в файл текст другой программы в соответствии с листингом 8.3. (рис. 2.10).



```
lab8-3.asm [-M--] 30 L:[ 1+21 22/ 29] *(10
#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'
```

Рис. 2.10: Файл листинга “lab8-3.asm”

Создаем новый исполняемый файл программы и запускаем его. Также проводим проверку. (рис. 2.11).

```
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-3 15 2 33 26
Результат: 76
aytueva@dk6n50 ~/work/arch-pc/lab08 $
```

Рис. 2.11: Запуск исполняемого файла

Изменяем в текст программы для вычисления произведения аргументов командной строки. (рис. 2.12).

```

lab8-3.asm      [-M--] 11 L:[ 1+2
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi, 1
next:
cmp ecx,0h
jz _end
pop eax
call atoi
mov ebx, esi
mul ebx
mov esi,eax
loop next
_end:
mov eax, msg
call sprint

```

Рис. 2.12: Редактирование файла

Создаем новый исполняемый файл программы и запускаем его. Также проводим проверку. (рис. 2.13).

```
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-3 3 5 2
Результат: 30
aytueva@dk6n50 ~/work/arch-pc/lab08 $
```

Рис. 2.13: Запуск исполняемого файла

3 Выполнение самостоятельной работы

Создаем новый файл “lab8-4.asm” с помощью команды “touch”.

Напишите программу, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$. Мой вариант - 11 (рис. 3.1).


```

lab8-4.asm      [-M++]
%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
_end:
mov eax,otv
call sprintf
mov eax,esi
call iprintLF
call quit

```

Рис. 3.1: Редактирование файла

Создаем исполняемый файл программы и запускаем его. Ответ верный. (рис. 3.2).

```
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-4
f(x)=15x+2
Результат: 0
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-4 2 3
f(x)=15x+2
Результат: 79
aytueva@dk6n50 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
aytueva@dk6n50 ~/work/arch-pc/lab08 $ ./lab8-4 2 3 4
f(x)=15x+2
Результат: 141
```

Рис. 3.2: Запуск исполняемого файла

4 Выводы

Благодаря данной лабораторной работе я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.