

Отчёт по лабораторной работе №8

дисциплина: Архитектура компьютера

Бражко Александра Александровна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выполнение самостоятельной работы	14
6	Выводы	16
	Список литературы	17

Список иллюстраций

4.1	Выполнение программ	8
4.2	Ввод текста программы	8
4.3	Создание и проверка	9
4.4	Изменение текста программы	9
4.5	Создание и проверка	9
4.6	Изменение текста программ	10
4.7	Создание и проверка	10
4.8	Создание файла	10
4.9	Ввод текста программы	11
4.10	Создание и проверка	11
4.11	Создание файла	11
4.12	Ввод текста программы	12
4.13	Создание и проверка	12
4.14	Изменение текста программ	13
4.15	Создание и проверка	13
5.1	Создание файла	14
5.2	Написание программы	14
5.3	Создание и проверка	15

Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . . .	7
-----	---	---

1 Цель работы

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

2 Задание

1. Реализация циклов в NASM
2. Обработка аргументов командной строки
3. Задание для самостоятельной работы

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую систему
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [1–4].

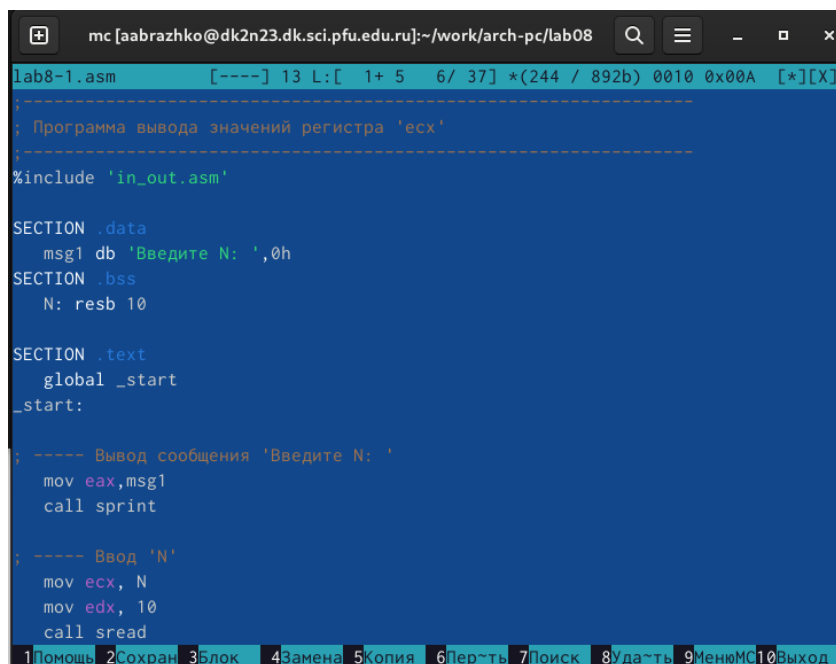
4 Выполнение лабораторной работы

Создаём каталог для программам лабораторной работы № 8, переходим в него и создаём файл lab8-1.asm (рис. 4.1).

```
aabrazhko@dk2n23 ~ $ mkdir ~/work/arch-pc/lab08
aabrazhko@dk2n23 ~ $ cd ~/work/arch-pc/lab08
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ touch lab8-1.asm
```

Рис. 4.1: Выполнение программ

Вводим в файл lab8-1.asm текст программы из листинга 8.1 (рис. 4.2).



```
lab8-1.asm  [----] 13 L: [ 1+ 5 6/ 37] *(244 / 892b) 0010 0x00A [*][X]
;-----
; 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

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход
```

Рис. 4.2: Ввод текста программы

Создаём исполняемый файл и проверяем его работу. (рис. 4.3).


```

aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $

```

Рис. 4.3: Создание и проверка

Изменим текст программы, добавив изменение значение регистра `ecx` в цикле (рис. 4.4).

```

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'
; переход на 'label'

```

Рис. 4.4: Изменение текста программы

Создаём исполняемый файл и проверяем его работу (рис. 4.5).

```

aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
7
5
3
1

```

Рис. 4.5: Создание и проверка

Внесим изменения в текст программы, добавив команды `push` и `pop` (рис. 4.6).

```

label:
    push ecx ; добавление значения ecx в стек
    sub ecx,1
    mov [N],ecx
    mov eax,[N]
    call iprintLF ; Вывод значения '\n'
    pop ecx ; извлечение значения ecx из стека
    loop label

```

Рис. 4.6: Изменение текста программ

Создаём исполняемый файл и проверяем его работу (рис. 4.7).

```

aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0

```

Рис. 4.7: Создание и проверка

Создаём файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 (рис. 4.8).

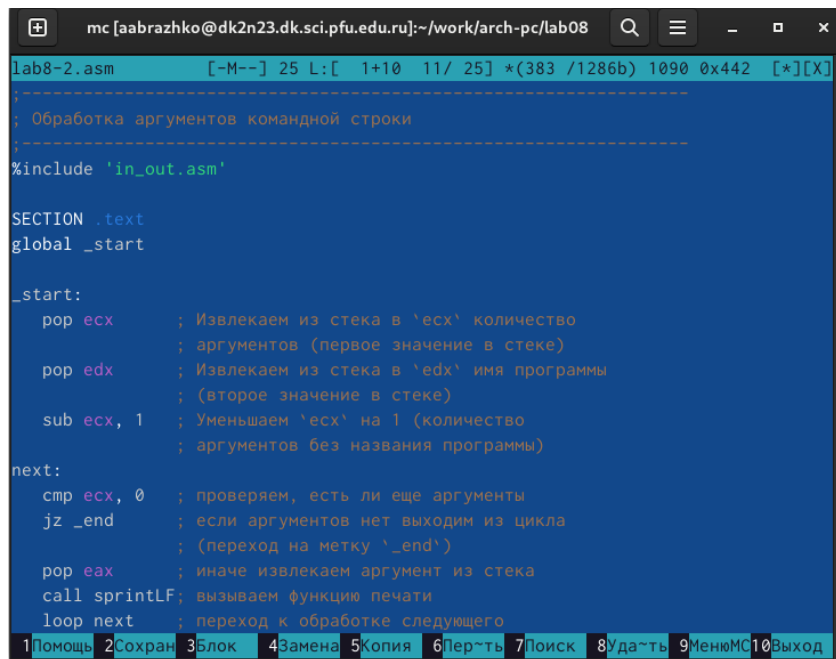
```

aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ touch lab8-2.asm

```

Рис. 4.8: Создание файла

Вводим в него текст программы из листинга 8.2 (рис. 4.9).



```
lab8-2.asm [-M--] 25 L: [ 1+10 11/ 25] *(383 /1286b) 1090 0x442 [*][X]
;-----
; Обработка аргументов командной строки
;-----
#include 'in_out.asm'

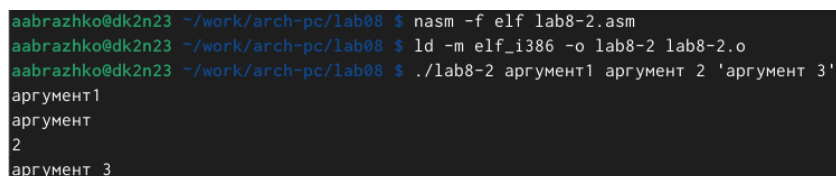
SECTION .text
global _start

_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

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход
```

Рис. 4.9: Ввод текста программы

Создаём исполняемый файл и запустим его, указав аргументы: `~$./lab8-2 аргумент1 аргумент 2 'аргумент 3` (рис. 4.10).



```
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
2
аргумент 3
```

Рис. 4.10: Создание и проверка

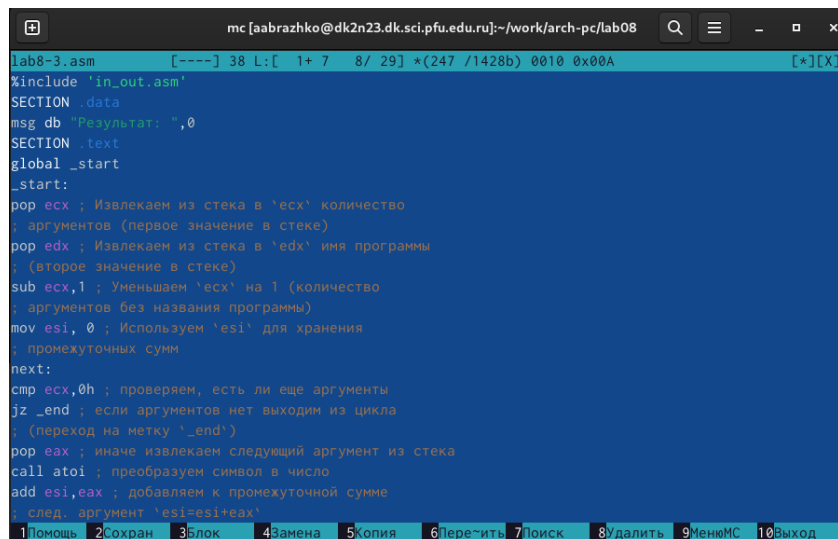
Создаём файл lab8-3.asm в каталоге `~/work/arch-pc/lab08` (рис. 4.11).



```
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ touch lab8-3.asm
```

Рис. 4.11: Создание файла

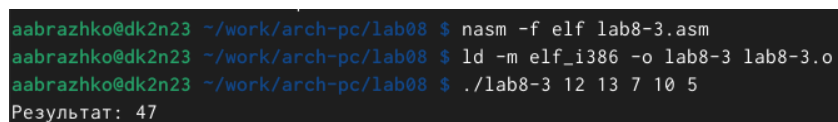
Вводим в него текст программы из листинга 8.3 (рис. 4.12).



```
lab8-3.asm [----] 38 L: [ 1+ 7 8/ 29] *(247 /1428b) 0010 0x00A [*][X]
#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. 4.12: Ввод текста программы

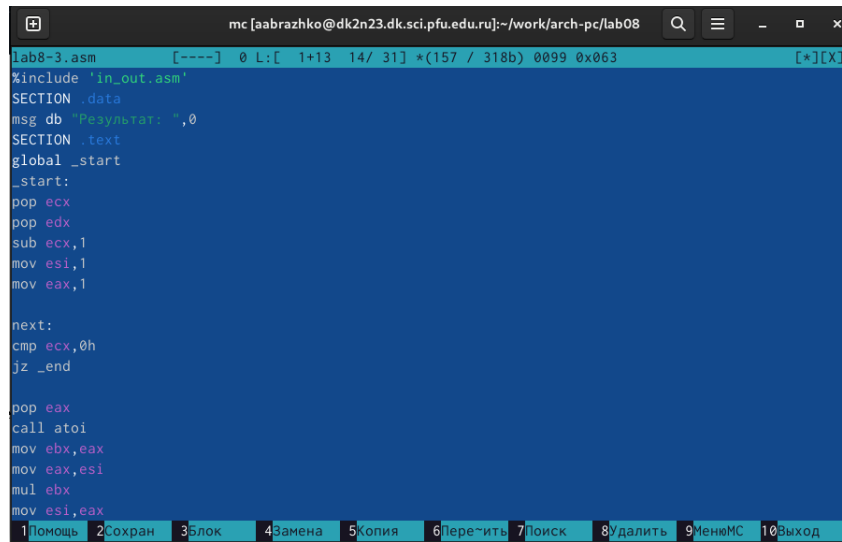
Создаём исполняемый файл и проверяем его работу, указав аргументы 2 13 7 10 5 (рис. 4.13).



```
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 47
```

Рис. 4.13: Создание и проверка

Изменим текст программы из листинга 8.3 для вычисления произведения аргументов командной строки (рис. 4.14).



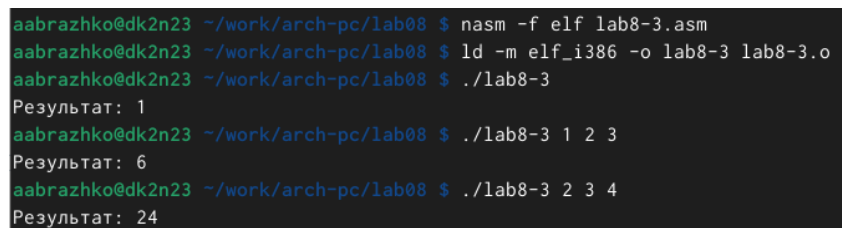
```
lab8-3.asm [----] 0 L: [ 1+13 14/ 31] *(157 / 318b) 0099 0x063 [*][X]
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi,1
mov eax,1

next:
cmp ecx,0h
jz _end

pop eax
call atoi
mov ebx,eax
mov eax,esi
mul ebx
mov esi,eax
```

Рис. 4.14: Изменение текста программ

Создаём исполняемый файл и проверяем его работу (рис. 4.15).



```
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-3
Результат: 1
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-3 1 2 3
Результат: 6
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-3 2 3 4
Результат: 24
```

Рис. 4.15: Создание и проверка

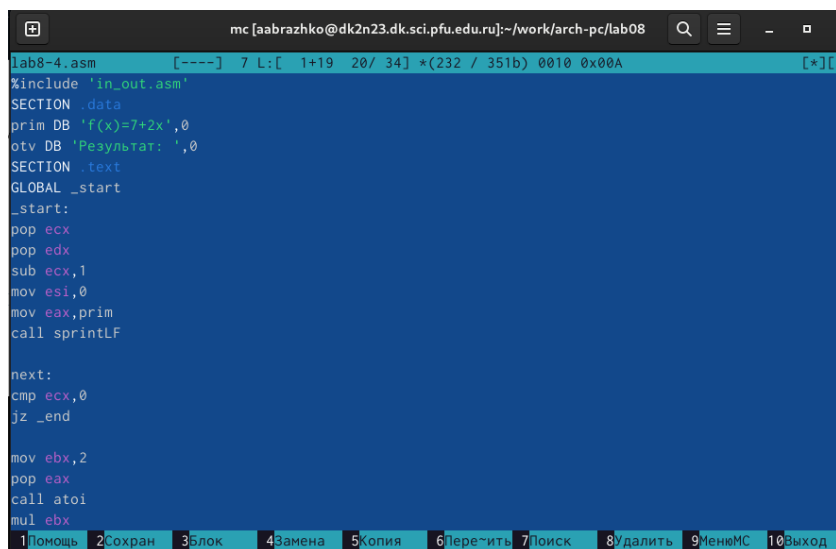
5 Выполнение самостоятельной работы

Создаём файл lab8-4.asm в каталоге ~/work/arch-pc/lab08 (рис. 5.1).

```
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ touch lab8-4.asm
```

Рис. 5.1: Создание файла

Напишем программу, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$. Функцию берём из таблицы 8.1 в соответствии с вариантом (у меня №8), полученным при выполнении лабораторной работы № 7 (рис. 5.2).



```
lab8-4.asm [-----] 7 L: [ 1+19 20/ 34] *(232 / 351b) 0010 0x00A [*][X]
#include 'in_out.asm'
SECTION .data
prim DB 'f(x)=7+2x',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,2
pop eax
call atoi
mul ebx
```

Рис. 5.2: Написание программы

Создайте исполняемый файл и проверьте его работу на нескольких наборах $x = x_1, x_2, \dots, x_n$. (рис. 5.3).

```
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4
f(x)=7+2x
Результат: 48
aabrazhko@dk2n23 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3
f(x)=7+2x
Результат: 33
```

Рис. 5.3: Создание и проверка

6 Выводы

В ходе выполнения работы я приобрела навыков написания программ с использованием циклов и обработкой аргументов командной строки.

Список литературы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.
2. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.