

Отчет по лабораторной работе №8

Дисциплина: архитектура компьютера

Гончарь Анастасия Александровна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Реализация циклов в NASM	8
4.2	Обработка аргументов командной строки	11
4.3	Задание для самостоятельной работы	14
4.3.1	Листинг для файла lab8-4.asm	15
5	Выводы	17
	Список литературы	18

Список иллюстраций

4.1	Создание файла и каталога	8
4.2	Текст программы	9
4.3	Запуск файла	9
4.4	Измененный текст программы	10
4.5	Запуск файла	10
4.6	Редактирование текста программы	10
4.7	Запуск файла	11
4.8	Создание файла	11
4.9	Текст программы	11
4.10	Запуск файла	12
4.11	Создание файла	12
4.12	Текст программы	12
4.13	Запуск файла	13
4.14	Текст программы	13
4.15	Запуск файла	13
4.16	Создание файла	14
4.17	Текст программы	14
4.18	Результаты работы программы	15

Список таблиц

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

1 Цель работы

Целью данной лабораторной работы является 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

4.1 Реализация циклов в NASM

Создаю каталог для лабораторной работы № 8, перехожу в него и создаю файл lab8-1.asm (рис. 4.1).

```
aagoncharj@dk3n33 ~ $ mkdir ~/work/arch-pc/lab08  
aagoncharj@dk3n33 ~ $ cd ~/work/arch-pc/lab08  
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ touch lab8-1.asm  
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $
```

Рис. 4.1: Создание файла и каталога

Открываю созданный файл и ввожу в него текст программы из листинга 8.1 (рис. 4.2).


```
lab8-1.asm      [-M--] 24 L:[ 1+ 5 6/ 31] *(275 / 844b) 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
; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход
```

Рис. 4.2: Текст программы

Создаю исполняемый файл и запускаю его (рис. 4.3).

```
-----
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 1
1
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 4
4
3
2
1
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ █
```

Рис. 4.3: Запуск файла

Изменяю текст программы, в теле цикла label добавляю строку `sub eax,1` (рис. 4.4).

```

lab8-1.asm      [-M--] 44 L: [ 10+20  30/ 32] *(828 / 868b) 0010 0x00A  [*][X]
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 ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'

```

Рис. 4.4: Измененный текст программы

Создаю исполняемый файл и запускаю его (рис. 4.5).

```

aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
7
5
3
1
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $

```

Рис. 4.5: Запуск файла

Теперь вношу изменения в текст программы добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop (рис. 4.6).

```

label:
push ecx ; добавление значения ecx в стек
sub ecx,1 ; 'ecx=ecx-1'
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
pop ecx ; извлечение значения ecx из стека
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit

```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9Меню 10Выход

Рис. 4.6: Редактирование текста программы

Создаю исполняемый файл и запускаю его (рис. 4.7).

```
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 6
5
4
3
2
1
0
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ █
```

Рис. 4.7: Запуск файла

4.2 Обработка аргументов командной строки

Создаю файл lab8-2.asm (рис. 4.8).

```
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ touch lab8-2.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ █
```

Рис. 4.8: Создание файла

Открываю файл и ввожу в него текст из листинга 8.2 (рис. 4.9).

```
lab8-2.asm  [-M--] 13 L:[ 1+ 4 5/ 23] *(243 /1151b) 0010 0x00A  [*][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 sprintLF ; вызываем функцию печати
loop next ; переход к обработке следующего
; аргумента (переход на метку 'next')
_end:
```

Рис. 4.9: Текст программы

Создаю исполняемый файл и запускаю его (рис. 4.10).

```
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-2 A n a s t a s i a
A
n
a
s
t
a
s
i
a
-
```

Рис. 4.10: Запуск файла

Далее создаю файл lab8-3.asm (рис. 4.11).

```
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ touch lab8-3.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $
```

Рис. 4.11: Создание файла

Открываю файл и ввожу в него текст из листинга 8.3 (рис. 4.12).

```
lab8-3.asm      [-M--]  7 L:[ 1+ 5  6/ 29] *(103 /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'
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход
```

Рис. 4.12: Текст программы

Создаю исполняемый файл и запускаю его (рис. 4.13).

```

aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 19 5
Результат: 56
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $

```

Рис. 4.13: Запуск файла

Изменяю программу так, чтобы она выводила произведение введенных чисел (рис. 4.14).

```

lab8-3.asm  [----]  0 L:[ 1+ 0 1/ 33] *(0 /1469b) 0037 0x025 [*][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,1 ; Используем 'esi' для хранения
    ; промежуточных сумм
    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 ; 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. 4.14: Текст программы

Создаю исполняемый файл и запускаю его (рис. 4.15).

```

aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ ./lab8-3 2 3 7 1 5
Результат: 210
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $

```

Рис. 4.15: Запуск файла

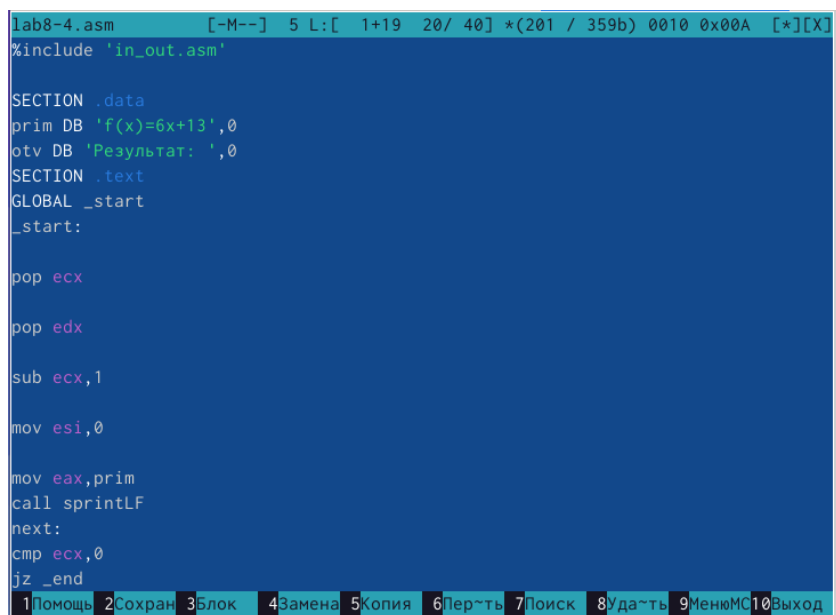
4.3 Задание для самостоятельной работы

Создаю файл lab8-4.asm (рис. 4.16).

```
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $ touch lab8-4.asm
aagoncharj@dk3n33 ~/work/arch-pc/lab08 $
```

Рис. 4.16: Создание файла

У меня 15 вариант, поэтому пишу программу для $f(x)=6x+13$ в созданном файле (рис. 4.17).



```
lab8-4.asm      [-M--]  5 L: [ 1+19  20/ 40] *(201 / 359b) 0010 0x00A  [*][X]
#include 'in_out.asm'

SECTION .data
prim DB 'f(x)=6x+13',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

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

Рис. 4.17: Текст программы

Создаю исполняемый файл и запускаю его (рис. 4.18). Программа работает верно.

```

aagoncharj@edk3n33 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
aagoncharj@edk3n33 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
aagoncharj@edk3n33 ~/work/arch-pc/lab08 $ ./lab8-4 4
f(x)=6x+13
Результат: 37
aagoncharj@edk3n33 ~/work/arch-pc/lab08 $ ./lab8-4 10
f(x)=6x+13
Результат: 73
aagoncharj@edk3n33 ~/work/arch-pc/lab08 $ ./lab8-4 7
f(x)=6x+13
Результат: 55
aagoncharj@edk3n33 ~/work/arch-pc/lab08 $

```

Рис. 4.18: Результаты работы программы

4.3.1 Листинг для файла lab8-4.asm

```

%include 'in_out.asm'

SECTION .data
prim DB 'f(x)=6x+13',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,6
pop eax
call atoi
mul ebx

add eax,13

add esi,eax

loop next

_end:
mov eax,otv
call sprint
mov eax,esi
call iprintLF
call quit
```


5 Выводы

При выполнении данной лабораторной работы я приобрела навыки 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.