

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ
НАРОДОВ**

**Факультет физико-математических и естественных наук
Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6**

дисциплина: *Архитектура компьютера*

Студент: Силантьева Анастасия

Группа: НКАбд-05-25

№ ст. билета: 1032253541

МОСКВА

2025 г.

Содержание

Список иллюстраций	3
1. Цель работы	4
2. Выполнение лабораторной работы	5
2.1 Работа с NASM5	
2.2 Выполнение арифметических операций в NASM	9
2.3 Задания для самостоятельной работы	14
3. Вывод	16
Список литературы	17

Список иллюстраций

1. Рис.1 Создание папки lab06 и файла lab6-1.asm 5
2. Листинг 6.1 Заполнение файла lab6-1.asm в соответствии с листингом 6
3. Рис. 2 Работа с файлом и его проверка 6
4. Создание папки «lab05». Рис. 3 Исправленный код 7
5. Рис. 4 Вывод исправленного кода 7
6. Рис. 5 Создание файла lab6-2.asm 8
7. Листинг 6.2 10
8. Рис. 6 Вывод кода
9. Рис. 7 Измененные строки в коде 13
- 10.Рис. 8 Вывод кода с измененными строками 14
- 11.Рис. 9 Код с изменением `iprintLF` на `iprint` 14
- 12.Рис. 10 Вывод кода 15
- 13.Рис. 11 Создание файла lab6-3.asm 16
- 14.Листинг 6.3 17
- 15.Рис. 12 Вывод программы lab6-3.asm 18
- 16.Рис. 13 Измененный текст программы lab6-3.asm 18
- 17.Рис. 14 Вывод измененной программы 19
- 18.Рис. 15 Создание файла variant.asm 19
19. Листинг 6.4
- 20.Рис. 16 Вывод файла variant.asm
- 21.Рис. 17 Программа для вычисления выражения из варианта 2
- 22.Рис. 18 Создание файла samost-2.asm
- 23.Рис. 19 Вывод программы для вычисления выражения из варианта 2

1. Цель работы

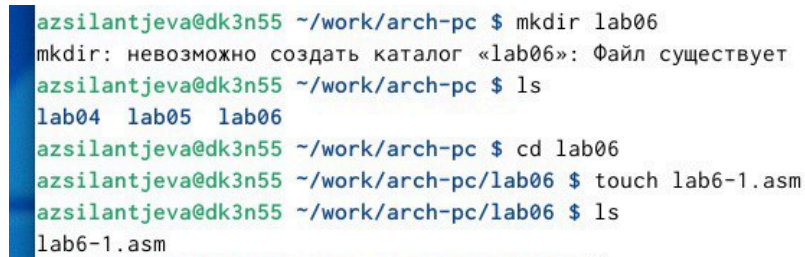
Освоение арифметических инструкций языка ассемблера NASM.

2. Выполнение лабораторной работы

2.1 Работа с NASM

Создайте каталог для программ лабораторной работы № 6, перейдите в него и создайте файл lab6-1.asm

Я создала каталог для программ лабораторной работы № 6, перешла в него и создала файл lab6-1.asm ([Рис. 1](#))



```
azsillantjeva@dk3n55 ~/work/arch-pc $ mkdir lab06
mkdir: невозможно создать каталог «lab06»: Файл существует
azsillantjeva@dk3n55 ~/work/arch-pc $ ls
lab04  lab05  lab06
azsillantjeva@dk3n55 ~/work/arch-pc $ cd lab06
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-1.asm
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ls
lab6-1.asm
```

Рис. 1 Создание папки lab06 и файла lab6-1.asm

Введите в файл lab6-1.asm текст программы из листинга 6.1. В данной программе в регистр `eax` записывается символ 6 (`mov eax,'6'`), в регистр `ebx` символ 4 (`mov ebx,'4'`). Далее к значению в регистре `eax` прибавляем значение регистра `ebx` (`add eax,ebx`, результат сложения запишется в регистр `eax`). Далее выводим результат. Так как для работы функции `sprintLF` в регистр `eax` должен быть записан адрес, необходимо использовать дополнительную переменную.

Я ввела в файл lab6-1.asm текст программы из [Листинга 6.1](#). В данной программе в регистр `eax` записывается символ 6 (`mov eax,'6'`), в регистр `ebx` символ 4 (`mov ebx,'4'`) и выполнила все действия.

```
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit
```

Листинг 6.1 Заполнение файла lab6-1.asm в соответствии с листингом

Создайте исполняемый файл и запустите его.

Создала и запустила файл ([Рис. 2](#))

```
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-1
j
```

Рис. 2 Работа с файлом и его проверка

Создайте исполняемый файл и запустите его. Как и в предыдущем случае при исполнении программы мы не получим число 10. В данном случае выводится символ с кодом 10. Отображается ли этот символ при выводе на экран?

Я исправила файл ([Рис. 3](#)) и запустила его ([Рис. 4](#)). При выводе на экран символ не отображается.

```
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit
```

Рис. 3 Исправленный код

```
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-1
```

Рис. 4 Вывод исправленного кода

Создайте файл *lab6-2.asm* в каталоге *~/work/arch-pc/lab06* и введите в него текст программы из листинга 6.2.

Создала файл *lab6-2.asm* (Рис. 5) в каталоге *~/work/arch-pc/lab06* и ввела в него текст программы из [листинга 6.2](#).

```
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm      lab6-1.asm.save.1  lab6-2.asm
```

Рис. 5 Создание файла *lab6-2.asm*

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF
call quit
|
```

Листинг 6.2

Создайте исполняемый файл и запустите его.

Создала исполняемый файл и запустила его (Рис. 6).

```
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-2
106
```

Рис. 6 Вывод кода

Далее я изменила строки в коде(Рис. 7) и вывела его (Рис. 8).

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Рис. 7 Измененные строки в коде

```
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-2
10
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $
```

Рис. 8 Вывод кода с измененными строками

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 9 Код с изменением iprintLF на iprint


```

azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-2
10azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $

```

Рис. 10 Вывод кода

2.2 Выполнение арифметических операций в NASM

*В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения $f(x) = (5 * 2 + 3)/3$.*

Создайте файл lab6-3.asm в каталоге ~/work/arch-pc/lab06. Внимательно изучите текст программы из листинга 6.3 и введите в lab6-3.asm.

Создала файл (Рис. 11) и ввела в него тест из [листинга 6.3](#).

```

azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-3.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm      lab6-1.asm.save.1  lab6-2      lab6-2.o
lab6-1      lab6-1.asm.save  lab6-1.o           lab6-2.asm  lab6-3.asm

```

Рис. 11 Создание файла lab6-3.asm

```

;-----
; Программа вычисления выражения
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,3 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,3 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
|

```

Листинг 6.3

```

azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ █

```

Рис. 12 Вывод программы lab6-3.asm

Изменила выражение на $f(x) = (4 * 6 + 2)/5$ и запустила код.

```

;-----
; Программа вычисления выражения
;-----
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,4 ; EAX=4
mov ebx,6 ; EBX=6
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+2
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=5
div ebx ; EAX=EAX/5, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

Рис. 13 Измененный текст программы lab6-3.asm

```

azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
azsillantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1

```

Рис. 14 Вывод измененной программы

Создайте файл `variant.asm` в каталоге `~/work/arch-pc/lab06` и внимательно изучите текст программы из листинга 6.4 и введите в файл `variant.asm`.

Создала файл ([Рис. 15](#)) и ввела [листинг 6.4](#).

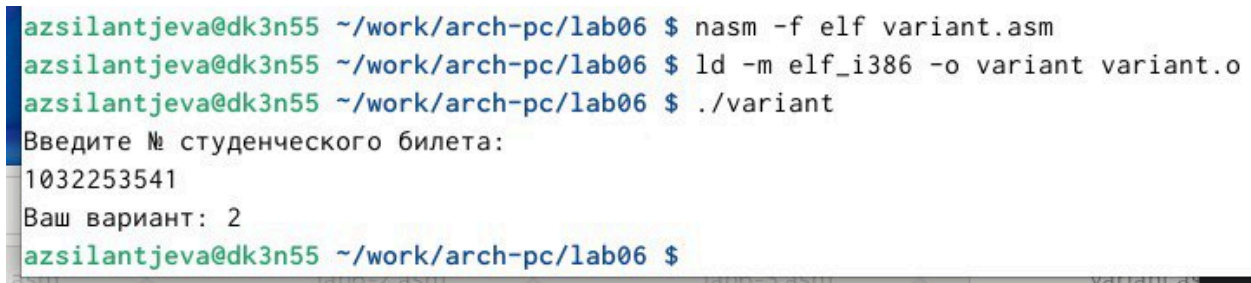
```
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ touch variant.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm.save  lab6-2  lab6-3  variant.asm
```

Рис. 15 Создание файла `variant.asm`

```
;-----
; Программа вычисления варианта
;-----
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, `eax=x`
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit
```

Листинг 6.4

Я запустила файл и был такой вывод ([Рис. 16](#)).



```
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1032253541
Ваш вариант: 2
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $
```

Рис. 16 Вывод файла variant.asm

Ответы на вопросы по листингу 6.4:

1. `mov eax,rem` и `call sprint`
2. Это подготовка к вызову `sread`: в `ecx` кладется адрес буфера `x`, в `edx` — его размер (80 байт). `call sread` читает строку с клавиатуры в этот буфер.
3. `call atoi` преобразует строку (символы), которую ввел пользователь, в число для арифметических операций.
4. Строки от `xor edx,edx` до `inc edx`.
5. Остаток от деления записывается в регистр `edx`.
6. `inc edx` увеличивает остаток на 1, чтобы получить вариант от 1 до 20 (т.к. остаток от деления на 20 будет от 0 до 19).
7. `mov eax,rem + call sprint` (вывод сообщения) и `mov eax,edx + call iprintLF` (вывод номера варианта).

2.3 Задания для самостоятельной работы

Написать программу вычисления выражения $y = f(x)$. Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений. Вид функции $f(x)$ выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы.

Вариант 2

```
%include 'in_out.asm'
SECTION .data
    div: DB 'Выражение для вычисления:  $y = (12x + 3) \times 5$ ',
    0
    msg: DB 'Введите x: ', 0
    rem: DB 'Результат вычисления равен: ', 0

SECTION .bss
    x resb 10          ; буфер для ввода x
    result resd 1       ; место для хранения результата

SECTION .text
GLOBAL _start
_start:
    ; Выводим выражение для вычисления
    mov eax, div
    call sprintLF

    ; Выводим запрос на ввод x
    mov eax, msg
    call sprint

    ; Читаем значение x
    mov ecx, x
    mov edx, 10
    call sread

    ; Преобразуем строку в число
    mov eax, x
    call atoi
    mov [x], eax

    ; Вычисляем  $12 \times x$ 
    mov eax, [x]        ; загружаем x в eax
    mov ebx, 12          ; ebx = 12
    imul ebx             ; eax = 12 * x
```

Рис. 17 Программа для вычисления выражения из варианта 2


```

azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf samost-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o samost-2 samost-2.o
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./samost-2
Выражение для вычисления:  $y = (12 * x + 3) * 5$ 
Введите x: 1
Результат вычисления равен: 75
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf samost-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o samost-2 samost-2.o
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ./samost-2
Выражение для вычисления:  $y = (12 * x + 3) * 5$ 
Введите x: 6
Результат вычисления равен: 375

```

Рис. 18 Создание файла samost-2.asm

```

azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ touch samost-2.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm.save  lab6-2  lab6-3  samost-2.asm  variant.o
lab6-1      lab6-1.asm.save.1 lab6-2.asm lab6-3.asm variant
lab6-1.asm  lab6-1.o          lab6-2.o  lab6-3.o  variant.asm
azsilantjeva@dk3n55 ~/work/arch-pc/lab06 $

```

Рис. 19 Вывод программы для вычисления выражения из варианта 2

3. Вывод

В ходе проведенной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.

Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. —
6. 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
7. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
8. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
9. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
10. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
11. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
12. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
13. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
14. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
15. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
16. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
17. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).