**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: Лебеденко Елена Викторвна

Группа: НКАбд-04-25

**МОСКВА**

2025 г.

# Содержание

1. Цель работы…….……4

2. Выполнение лабораторной работы…….……..5-11

2.1 Символьные и численные данные в NASM..…..……5

2.2 Выполнение арифметических операций в NASM..……..…….8

Ответы на вопросы……..11-12

3. Задания для самостоятельной работы……...12-14

4. Вывод……..……15

# Список иллюстраций

1.1 Создание каталога и файла для выполнения л/р……....5

1.2 Программа из листинга 6.1 в редакторе…….….5

1.3 Вывод программы из листинга 6.1…..…….……....5

1.4 Изменённая программа из листинга 6.1...……...6

1.5 Вывод изменённой программы.……...6

1.6 Программа из листинга 6.2…….……..……7

1.7 Вывод программы из листинга 6.2...……7

1.8 Изменённая программа из листинга 6.2.……7

1.9 Вывод изменённой программы….……...…8

1.10 iprint вместо iprintLF……..…….……8

2.1 Файл lab6-3.asm……....……...…….8

2.2 Программа из листинга 6.3…….…….....9

2.3 Вывод программы из листинга 6.3….……...……...9

2.4 Программа для $f(x) = (4 * 6 + 2)/5$……...9

2.5 Вывод изменённой программы.……..……...10

2.6 Файл variant.asm……..……..……...10

2.7 Программа из листинга 6.4……..……...10

2.8 Вывод программы из листинга 6.4……...……..……....11

3.1 Задание…….……..……...12

3.2 Задание (для x1 и x2)……..……..……...12

# 1. Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2. Выполнение лабораторной работы

## 2.1. Символьные и численные данные в NASM

1) Создаю каталог для программам лабораторной работы № 6, перехожу в него и создаю файл lab6-1.asm:

```
evlebedenko@evlebedenko:~$ mkdir ~/work/arch-pc/lab06
evlebedenko@evlebedenko:~$ cd ~/work/arch-pc/lab06
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ touch lab6-1.asm
```

1.1 Создание каталога и файла для выполнения л/р

2) Ввожу в файл lab6-1.asm текст программы из листинга 6.1.

```
                                evle
  GNU nano 7.2
%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
```

1.2 Программа из листинга 6.1 в редакторе

Создаю исполняемый файл и запускаю его.

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

1.3 Вывод программы из листинга 6.1

В данном случае при выводе значения регистра eax мы ожидаем увидеть число 10. Однако результатом будет символ j. Это происходит потому, что код символа

6 равен 00110110 в двоичном представлении (или 54 в десятичном представлении), а код символа 4 – 00110100 (52). Команда add eax,ebx запишет в регистр eax сумму кодов – 01101010 (106), что в свою
очередь является кодом символа j.

3) Теперь изменим текст программы и вместо символов, запишем в регистры числа. Исправляю текст программы следующим образом: убираю кавычки в строках, где записываю числа в переменные.

```
 GNU nano 7.2
%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
```

1.4 Изменённая программа из листинга 6.1

Создаю исполняемый файл и запускаю его.

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

```

1.5 Вывод изменённой программы

На это раз программы выдала пустую строчку. По таблице ASCII 10 — то переход на новую строку.

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

```
 GNU nano 7.2
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

1.6 Программа из листинга 6.2

Создаю исполняемый файл и запускаю его.

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

1.7 Вывод программы из листинга 6.2

В результате работы программы мы получим число 106. В данном случае, как и в первом, команда add складывает коды символов ‘6’ и ‘4’ (54+52=106). Однако, в отличии от программы из листинга 6.1, функция iprintLF позволяет вывести число, а не символ, кодом которого является это число.

5) Аналогично предыдущему примеру изменим символы на числа. Уберу кавычки из строк, в которых записываю в переменные числа

```
 GNU nano 7.2
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

1.8 Изменённая программа из листинга 6.2

Создаю исполняемый файл и запускаю его.

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

1.9 Вывод изменённой программы

При исполнение программы мы получаем 10.

Если заменить функцию iprintLF на iprint, ответ выведется на той же строке.

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

1.10 iprint вместо iprintLF

T.e. iprint оставляет курсор на той же строке, а iprintLF переносит его на следующую.

## 2.2 Выполнение арифметических операций в NASM

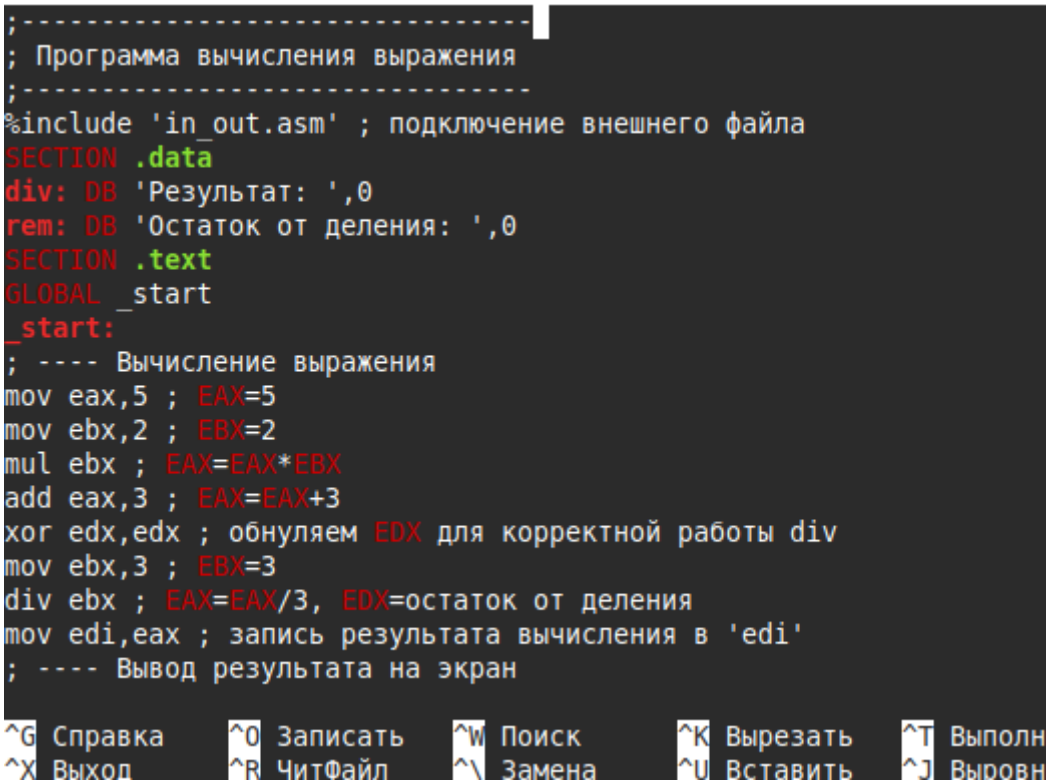
6) В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения $f(x) = (5 * 2 + 3)/3$.

Создаю файл lab6-3.asm в каталоге ~/work/arch-pc/lab06:

```
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
```

2.1 Файл lab6-3.asm

Изучаю текст программы из листинга 6.3 и ввожу в lab6-3.asm.




2.2 Программа из листинга 6.3

Создаю исполняемый файл и запускаю его.

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

2.3 Вывод программы из листинга 6.3

Результат совпал с примером

Изменяю текст программы для вычисления выражения $f(x) = (4 * 6 + 2)/5$, создаю исполняемый файл и проверяю его работу.

```
; ---- 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'
```

2.4 Программа для $f(x) = (4 * 6 + 2)/5$

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

2.5 Вывод изменённой программы

7) В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму:

• вывести запрос на введение № студенческого билета

• вычислить номер варианта по формуле: $(Sn \bmod 20) + 1$, где $Sn$ – номер студенческого билета (В данном случае $a \bmod b$ – это остаток от деления $a$ на $b$).

• вывести на экран номер варианта

Создаю файл variant.asm в каталоге ~/work/arch-pc/lab06:

```
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
```

2.6 Файл variant.asm

Изучаю текст программы из листинга 6.4 и ввожу в файл variant.asm.

```
 GNU nano 7.2                                  variant.asm *
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
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
```

2.7 Программа из листинга 6.4

Создаю исполняемый файл и запускаю его.

```
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ nano variant.asm
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ nasm -f elf variant.asm
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032253535
Ваш вариант: 16
```

2.8 Вывод программы из листинга 6.4

## Ответы на вопросы:

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

```
mov eax,rem
call sprint
```

2. Для чего используется следующие инструкции?

```
mov  ecx, x
mov  edx, 80
call sread
```

*mov ecx, x*

Загружает в регистр ecx адрес переменной x.

*mov edx, 80*

Загружает в регистр edx число 80 — максимальное количество байт, которые можно считать.

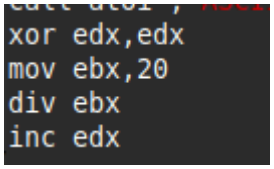
*call sread*

Вызывает подпрограммы из внешнего файла, обеспечивает ввод сообщения с клавиатуры.

3. Для чего используется инструкция “call atoi”?

Вызывает подпрограмму из внешнего файла. Она преобразует код в целое число и записывает его в регистр eax.

4. Какие строки листинга 6.4 отвечают за вычисления варианта?



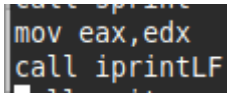

5. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?

edx

6. Для чего используется инструкция “inc edx”?

Увеличивает значение регистра на один

7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?




## 3. Задание для самостоятельной работы

1) 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. Создайте исполняемый файл и проверьте его работу для значений $x1$ и $x2$ из 6.3.

Буду работать с функцией: (10$x$ − 5)^2

```
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ nasm -f elf task.asm
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ld -m elf_i386 -o task task.o
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ./task
Значение x : 4
Реультат: 1225
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ./task
Значение x : 5
Реультат: 2025
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ./task
Значение x : 8
Реультат: 5625
```

3.1 Задание

```
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ./task
Значение x : 1
Реультат: 25
evlebedenko@evlebedenko:~/work/arch-pc/lab06$ ./task
Значение x : 2
Реультат: 225
```

3.2 Задание (для x1 и x2)

## *Листинг:*

```
%include 'in_out.asm' ;
SECTION .data
msg: DB 'Значение x : ',0
rem: DB 'Реультат: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,msg
call sprint
mov ecx,x
mov edx,80
call sread
```

```
mov eax, x
call atoi
mov ebx,10
mul ebx
sub eax,5
mul eax
mov edi,eax
mov eax,rem
call sprint
mov eax, edi
call iprintLF
call quit
```

# 4. Вывод

При выполнении лабораторной работы я освоила базовые арифметические инструкции языка ассемблера NASM.