
Front matter

title: "Лабораторная работа №6" subtitle: "Арифметические операции в NASM." author: "Арсоева
Залина НБИбд-01-21"

Generic options

lang: ru-RU toc-title: "Содержание"

Bibliography

bibliography: bib/cite.bib csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

PDF output format

toc: true # Table of contents toc-depth: 2 lof: true # List of figures lot: true # List of tables fontsize: 12pt
linestretch: 1.5 papersize: a4 documentclass: scrreprt

LaTeX polyglossia

polyglossia-lang: name: russian options: - spelling=modern - babelshorthands=true polyglossia-otherlangs:
name: english

LaTeX babel

babel-lang: russian babel-otherlangs: english

Fonts

mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions:
Ligatures=TeX romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase
monofontoptions: Scale=MatchLowercase,Scale=0.9

BibLaTeX

biblatex: true biblio-style: "gost-numeric" biblatexoptions:

- parenttracker=true
- backend=biber
- hyperref=auto
- language=auto
- autolang=other*
- citestyle=gost-numeric

Pandoc-crossref LaTeX customization

figureTitle: "Рис." tableTitle: "Таблица" listingTitle: "Листинг" lofTitle: "Список иллюстраций" lotTitle:
"Список таблиц" lolTitle: "Листинги"

Misc options

indent: true header-includes:

- `\usepackage{indentfirst}`
- `\usepackage{float} # keep figures where there are in the text`
- `\floatplacement{figure}{H} # keep figures where there are in the text`

Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

Задание

Символьные и численные данные в NASM

1. Создайте каталог для программ лабораторной работы No 7, перейдите в него и создайте файл `lab7-1.asm`:

```
mkdir ~/work/arch-pc/lab07 cd ~/work/arch-pc/lab07 touch lab7-1.asm
```

2. Введите в файл `lab7-1.asm` текст программы из листинга 7.1.

В данном случае при выводе значения регистра `eax` мы ожидаем увидеть число 10. Однако результатом будет символ `j`.

3. Далее изменим текст программы и вместо символов, запишем в регистры числа.

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. Пользуясь таблицей ASCII определите какому символу соответствует код 10. Отображается ли этот символ при выводе на экран?

Создайте файл `lab7-2.asm` в каталоге `~/work/arch-pc/lab07` и введите в него текст программы из листинга 7.2.

```
touch ~/work/arch-pc/lab07/lab7-2.asm
```

Создайте исполняемый файл и запустите его.

```
nasm -f elf lab7-1.asm ld -m elf_i386 -o lab7-1 lab7-1.o ./lab7-1
```

В результате работы программы мы получим число 106. В данном случае, как и в первом, команда `add` складывает коды символов `'б'` и `'4'` ($54+52=106$).

5. Аналогично предыдущему примеру изменим символы на числа.

Создайте исполняемый файл и запустите его. Какой результат будет получен при исполнении программы?

6. Создайте файл lab7-3.asm в каталоге ~/work/arch-pc/lab07:

touch ~/work/arch-pc/lab07/lab7-3.asm

Создайте исполняемый файл и запустите его. Результат работы программы должен быть следующим:

./lab7-3 Результат: 4 Остаток от деления: 1

Измените текст программы для вычисления выражения $f(x) = (4 * 6 + 2)/5$. Создайте исполняемый файл и проверьте его работу.

7. Создайте файл variant.asm в каталоге ~/work/arch-pc/lab07:

touch ~/work/arch-pc/lab07/variant.asm

Создайте исполняемый файл и запустите его. Проверьте результат работы программы вычислив номер варианта аналитически.

Включите в отчет по выполнению лабораторной работы ответы на следующие вопросы:

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения 'Ваш вариант:'?
2. Для чего используется следующие инструкции? `mov ecx, x` `mov edx, 80` `call sread`
3. Для чего используется инструкция "call atoi"?
4. Какие строки листинга 7.4 отвечают за вычисления варианта?
5. В какой регистр записывается остаток от деления при выполнении инструкции "div ebx"?
6. Для чего используется инструкция "inc edx"?
7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

Задание для самостоятельной работы

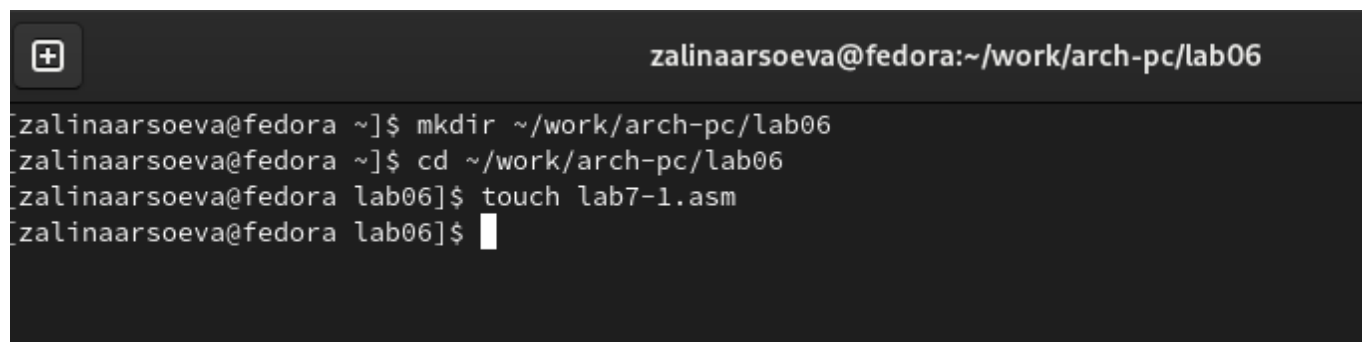
1. Написать программу вычисления выражения $y = f(x)$. Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений. Вид функции $f(x)$ выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создайте исполняемый файл и проверьте его работу для значений x_1 и x_2 из 6.3

Выполнение лабораторной работы

1. Создаю каталог для программ лабораторной работы No 7, перехожу в него и создаю файл lab7-1.asm:

mkdir ~/work/arch-pc/lab07 cd ~/work/arch-pc/lab07 touch lab7-1.asm

(рис. [-@fig:001])

A terminal window with a dark background. The title bar shows a plus icon and the text 'zalinaarsoeva@fedora:~/work/arch-pc/lab06'. The terminal content shows the following commands and their outputs:

```
zalinaarsoeva@fedora ~]$ mkdir ~/work/arch-pc/lab06
zalinaarsoeva@fedora ~]$ cd ~/work/arch-pc/lab06
zalinaarsoeva@fedora lab06]$ touch lab7-1.asm
zalinaarsoeva@fedora lab06]$
```

{ #fig:001 width=70% }

2. Введите в файл lab7-1.asm текст программы из листинга 7.1.

(рис. [-@fig:002])

A terminal window with a dark background. The title bar shows the text 'zalinaarsoeva@fedora lab06'. The terminal content shows the following commands and their outputs:

```
[zalinaarsoeva@fedora lab06]$ gedit lab7-1.asm
[zalinaarsoeva@fedora lab06]$ cat lab7-1.asm
%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
[zalinaarsoeva@fedora lab06]$
```

{ #fig:002 width=70% }

В данном случае при выводе значения регистра eax мы ожидаем увидеть число 10. Однако результатом будет символ j.

(рис. [-@fig:003])

```
[zalinaarsoeva@fedora lab06]$ ls
in_out.asm  lab7-1.asm
[zalinaarsoeva@fedora lab06]$ nasm -f elf lab7-1.asm
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[zalinaarsoeva@fedora lab06]$ ./lab7-1
j
[zalinaarsoeva@fedora lab06]$
```

{ #fig:003

width=70% }

3. Далее изменим текст программы и вместо символов, запишем в регистры числа.

(рис. [-@fig:004])

```
[zalinaarsoeva@fedora lab06]$ gedit lab7-1.asm
[zalinaarsoeva@fedora lab06]$ cat lab7-1.asm
%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
[zalinaarsoeva@fedora lab06]$
```

{ #fig:004 width=70% }

Создадим исполняемый файл и запустим его. Как и в предыдущем случае при исполнении программы мы не получим число 10. В данном случае выводится символ с кодом 10. (рис. [-@fig:005])

```
[zalinaarsoeva@fedora lab06]$ nasm -f elf lab7-1.asm
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-1 lab7-1.o
ld: невозможно найти lab7-1.0: Нет такого файла или каталога
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[zalinaarsoeva@fedora lab06]$ ./lab7-1
```

```
[zalinaarsoeva@fedora lab06]$
```

{ #fig:005

width=70% }

Создадим файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 и введем в него текст программы из листинга 7.2.

touch ~/work/arch-pc/lab07/lab7-2.asm

(рис. [-@fig:006])

```
[zalinaarsoeva@fedora lab06]$ touch lab7-02.asm
[zalinaarsoeva@fedora lab06]$ gedit lab7-02.asm
[zalinaarsoeva@fedora lab06]$ cat lab7-02.asm
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

    mov eax, '6'
    mov ebx, '4'
    call iprintLF

    call quit

[zalinaarsoeva@fedora lab06]$
```

{ #fig:006 width=70% }

Создайте исполняемый файл и запустите его.

nasm -f elf lab7-1.asm ld -m elf_i386 -o lab7-1 lab7-1.o ./lab7-1

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).

(рис. [-@fig:007])

```
54
[zalinaarsoeva@fedora lab06]$ gedit lab7-02.asm
[zalinaarsoeva@fedora lab06]$ nasm -f elf lab7-02.asm
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-02 lab7-02.o
[zalinaarsoeva@fedora lab06]$ ./lab7-02
106
[zalinaarsoeva@fedora lab06]$
```

{ #fig:007 width=70% }

5. Аналогично предыдущему примеру изменим символы на числа. рис. [-@fig:008])



```

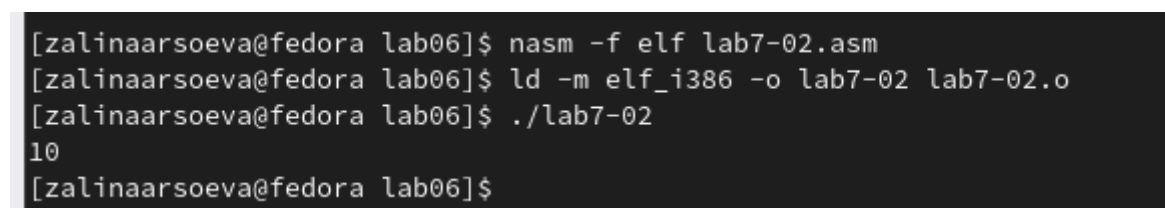
Открыть ▾  + *lab7-02.asm
~/work/arch-pc/lab06
1 %include 'in_out.asm'
2
3 SECTION .text
4 GLOBAL _start
5 _start:
6
7 mov eax, 6
8 mov ebx, 4
9 add eax,ebx
10 call iprintLF
11
12 call quit
13

```

{ #fig:008

width=70% }

Создадим исполняемый файл и запустим его. (рис. [-@fig:009])



```

[zalinaarsoeva@fedora lab06]$ nasm -f elf lab7-02.asm
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-02 lab7-02.o
[zalinaarsoeva@fedora lab06]$ ./lab7-02
10
[zalinaarsoeva@fedora lab06]$

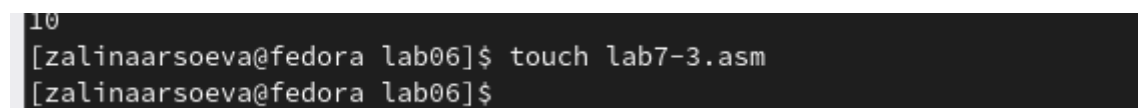
```

{ #fig:008

width=70% }

6. Создадим файл lab7-3.asm в каталоге ~/work/arch-pc/lab07:

touch ~/work/arch-pc/lab07/lab7-3.asm (рис. [-@fig:010])



```

10
[zalinaarsoeva@fedora lab06]$ touch lab7-3.asm
[zalinaarsoeva@fedora lab06]$

```

{ #fig:010

width=70% }

Создайте исполняемый файл и запустите его. Результат работы программы должен быть следующим:

./lab7-3 Результат: 4 Остаток от деления: 1

(рис. [-@fig:011]) (рис. [-@fig:012])

```

lab7-3.asm
~/work/arch-pc/lab06

1 %include 'in_out.asm'
2
3 SECTION .data
4
5 div: DB 'Результат: ', 0
6 rem: DB 'Остаток от деления: ', 0
7
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12 mov eax,5
13 mov ebx,2
14 mul ebx
15 add eax,3
16 xor edx, edx
17 mov ebx,3
18 div ebx
19
20 mov edi,eax
21
22 mov eax,div
23 call sprint
24 mov eax,edi
25 call iprintLF
26
27 mov eax,rem
28 call sprint
29 mov eax,edx
30 call iprintLF
31
32 call quit

```

{ #fig:011 width=70% }

```

[zalinaarsoeva@fedora lab06]$ touch lab7-3.asm
[zalinaarsoeva@fedora lab06]$ gedit lab7-3.asm
[zalinaarsoeva@fedora lab06]$ nasm -f elf lab7-3.asm
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-3 lab-3.0
ld: невозможно найти lab-3.0: Нет такого файла или каталога
[zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[zalinaarsoeva@fedora lab06]$ ./lab7-3
Результат: 4
Остаток от деления: 1
[zalinaarsoeva@fedora lab06]$

```

{ #fig:012 width=70% }

Изменим текст программы для вычисления выражения $f(x) = (4 * 6 + 2)/5$.

(рис. [-@fig:013])


```
zalinaarsoeva@fedora lab06]$ gedit lab7-3.asm
zalinaarsoeva@fedora lab06]$ cat lab7-3.asm
%include 'in_out.asm'

SECTION .data
div: DB 'Результат: ', 0
rem: DB 'Остаток от деления: ', 0

SECTION .text
GLOBAL _start
_start:

mov eax,4
mov ebx,6
mul ebx
add eax,2
xor edx, edx
mov ebx,5
div ebx

mov edi,eax

mov eax,div
call sprint
mov eax,edi
call iprintLF

mov eax,rem
call sprint
mov eax,edx
call iprintLF

call quit
zalinaarsoeva@fedora lab06]$ nasm -f elf lab7-3.asm
zalinaarsoeva@fedora lab06]$ ld -m elf_i386 -o lab7-3 lab7-3.o
zalinaarsoeva@fedora lab06]$ ./lab7-3
Результат: 5
Остаток от деления: 1
zalinaarsoeva@fedora lab06]$
```

#fig:013

width=70% }

7. Создадим файл variant.asm в каталоге ~/work/arch-pc/lab07:

touch ~/work/arch-pc/lab07/variant.asm

(рис. [-@fig:014])

```

variant.asm
~/work/arch-pc/lab06

1 %include 'in_out.asm'
2
3 SECTION .data
4 msg: DB 'Введите № студенческого билета: ', 0
5 rem: DB 'Ваш вариант: ', 0
6
7 SECTION .bss
8 x: RESB 80
9
10 SECTION .text
11 GLOBAL _start
12 _start:
13
14 mov eax, msg
15 call sprintLF
16
17 mov ecx, x
18 mov edx, 80
19 call sread
20
21 mov eax, x
22 call atoi
23
24 xor edx, edx
25 mov ebx, 20
26 div ebx
27 inc edx
28
29 mov eax, rem
30 call sprint
31 mov eax, edx
32 call iprintLF
33
34 call quit

```

{ #fig:014 width=70% }

Создаю исполняемый файл и запускаю его.

(рис. [-@fig:015])

```

[zalinaarsoeva@fedora lab06]$ ./variant
Введите № студенческого билета:
1132210611
Ваш вариант: 12
[zalinaarsoeva@fedora lab06]$

```

{ #fig:015
width=70% }

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения 'Ваш вариант:'?

mov eax,rem call sprint

2. Для чего используются следующие инструкции? `mov ecx, x` `mov edx, 80` `call sread`

3. Для чего используется инструкция “`call atoi`”?

вызов подпрограммы преобразования ASCII кода в число

4. Какие строки листинга 7.4 отвечают за вычисления варианта?

```
xor edx,edx mov ebx,20 div ebx inc edx
```

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

EDX

6. Для чего используется инструкция “`inc edx`”?

для прибавления на одну единицу

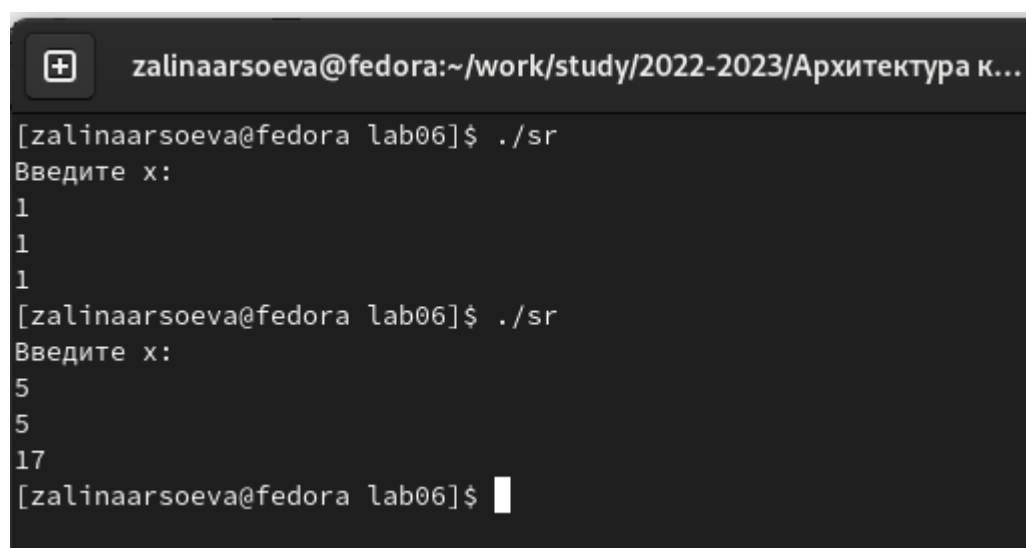
7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

```
mov eax,rem call sprint mov eax,edx call iprintLF
```

Выполнение задания для самостоятельной работы

1. Написала 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.

(рис. [-@fig:016])



```
zalinaarsoeva@fedora:~/work/study/2022-2023/Архитектура к...
[zalinaarsoeva@fedora lab06]$ ./sr
Введите x:
1
1
1
[zalinaarsoeva@fedora lab06]$ ./sr
Введите x:
5
5
17
[zalinaarsoeva@fedora lab06]$
```

{ #fig:016 width=70% }

2. Создала исполняемый файл и проверила его работу для значений x_1 и x_2 из 6.3

(рис. [-@fig:017])

```
[zalinaarsoeva@fedora lab06]$ cat sr.asm
%include 'in_out.asm'

SECTION .data
msg: DB 'Введите x: ', 0
rem: DB 'Результат: ', 0

SECTION .bss
x: RESB 80
res: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax, msg
call sprintLF

mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi

mov ebx, 8
mul ebx,
sub eax, 6
mov ebx, 2
div ebx
mov [res], eax

mov eax, x
call sprint

mov eax, [res]
call iprintLF

call quit
[zalinaarsoeva@fedora lab06]$
```

{ #fig:017 width=70% }

Выводы

Я освоила арифметических инструкций языка ассемблера NASM.