

Отчёта по лабораторной работе №6

НПИо6-03-23

Махмудов Суннатилло Баходир угли

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	19

Список иллюстраций

4.1	Создание деректории	8
4.2	файл lab6-1.asm	9
4.3	текст программы из листинга 6.1	9
4.4	исполняемый файл	10
4.5	изменил текст	10
4.6	исполняемый файл	10
4.7	файл lab6-2.asm	11
4.8	файл lab6-2.asm 106	11
4.9	изменим символы на числа	11
4.10	Создал исполняемый файл	12
4.11	Редактирование файла	12
4.12	Запуск исполняемого файла	12
4.13	lab6-3.asm	12
4.14	lab6-3.asm	13
4.15	Я ввел текст	13
4.16	Результат работы	14
4.17	Изменил текст	14
4.18	Проверял его работу	14
4.19	Проверял его работу	15
4.20	Проверял его работу	15
4.21	Проверял его работу	17

Список таблиц

1 Цель работы

Цель данного шаблона — максимально упростить подготовку отчётов по лабораторным работам. Модифицируя данный шаблон, студенты смогут без труда подготовить отчёт по лабораторным работам, а также познакомиться с основными возможностями разметки Markdown.

2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

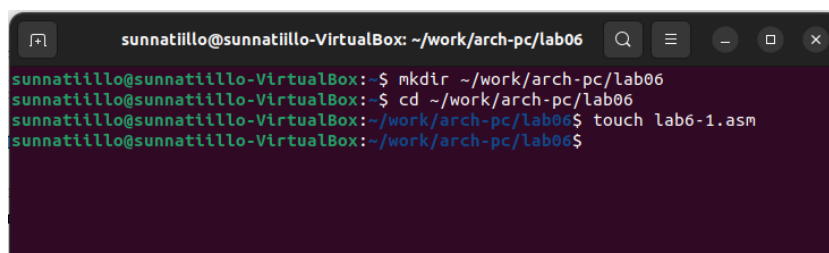
3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. Далее рассмотрены все существующие способы задания адреса хранения операндов – способы адресации.

4 Выполнение лабораторной работы

#Символьные и численные данные в NASM

Создаю каталог для программ лабораторной работы № 6, перейду в него и создаю файл lab6-1.asm:(рис. 4.21).



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

Рис. 4.1: Создание деректории

Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения записанные в регистр eax. (рис. 4.2).

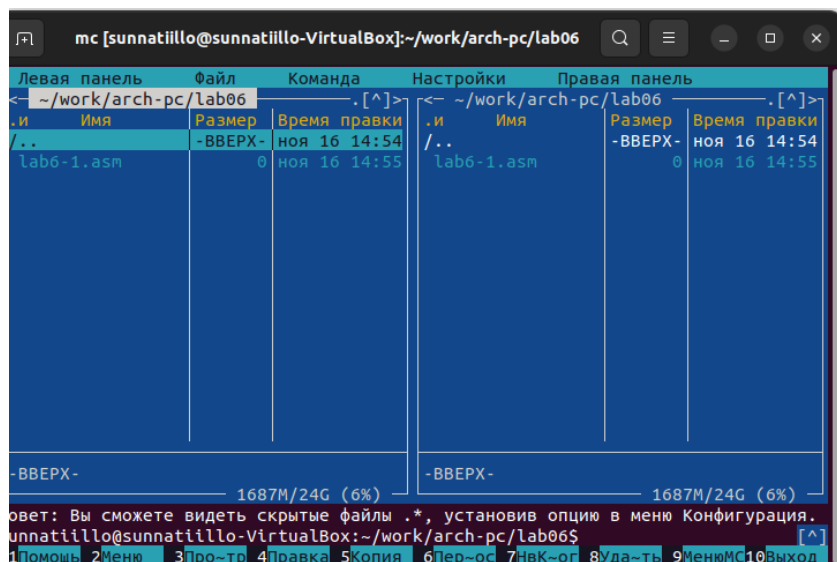


Рис. 4.2: файл lab6-1.asm

Введу в файл lab6-1.asm текст программы из листинга 6.1.(рис. 4.3).

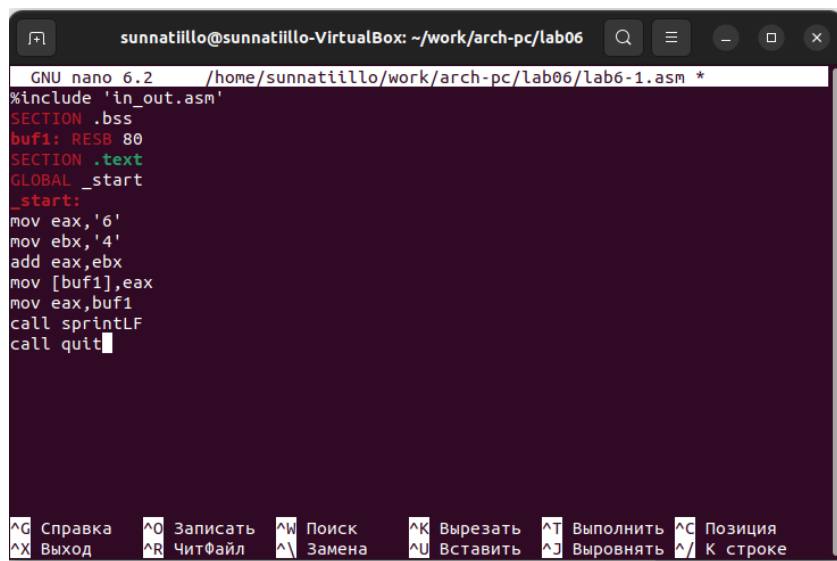


Рис. 4.3: текст программы из листинга 6.1

Создаю исполняемый файл и запускаю его.(рис. 4.4).

```

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

```

Рис. 4.4: исполняемый файл

Далее изменяю текст программы и вместо символов, запишу в регистры числа. Ис- правляю текст программы (Листинг 6.1) следующим образом: заменяю строки (рис. 4.5).

```

GNU nano 6.2 /home/sunnatillo/work/arch-pc/lab06/lab6-1.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 sprintf
call quit

```

Рис. 4.5: изменил текст

Создаю исполняю файл и запускаю его (рис. ??).

```

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

```

Рис. 4.6: исполняемый файл

Создаю файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и ввожу в него текст про- граммы из листинга 6.2.(рис. 4.7).

```

sunnatillo@sunnatillo-VirtualBox: ~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-2.asm
sunnatillo@sunnatillo-VirtualBox: ~/work/arch-pc/lab06$

```

Рис. 4.7: файл lab6-2.asm

Создаю исполняемый файл и запускаю его.(рис. 4.8).

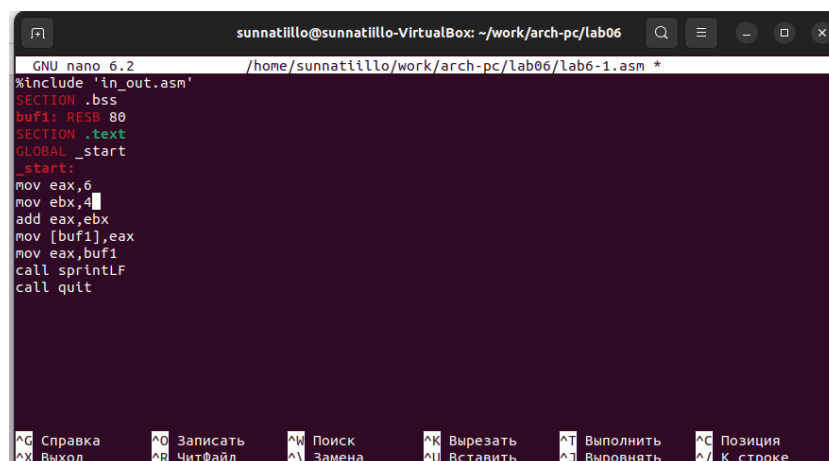
```

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

```

Рис. 4.8: файл lab6-2.asm 106

Аналогично предыдущему примеру изменяю символы на числа. Заменяю строки (рис. 4.9).



```

GNU nano 6.2 /home/sunnatillo/work/arch-pc/lab06/lab6-1.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 sprintf
call quit

```

Рис. 4.9: изменим символы на числа

Создаю исполняемый файл и запускаю его. Какой результат будет получен при исполнении программы?(рис. 4.10).

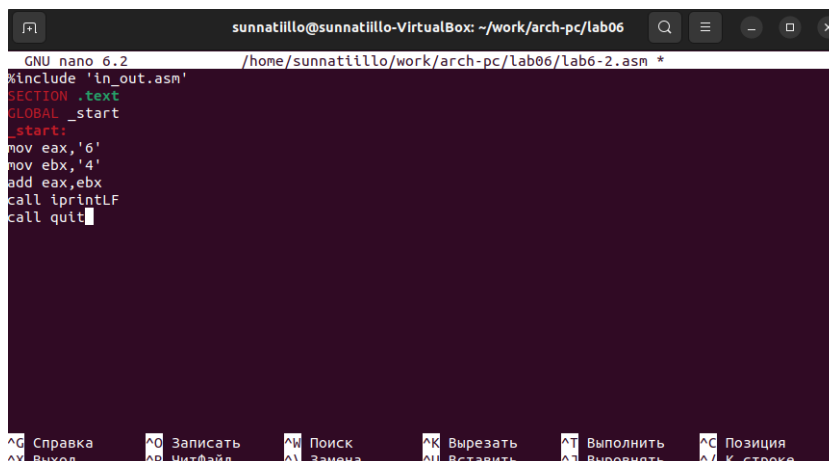
```

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

```

Рис. 4.10: Создал исполняемый файл

Заменяю в тексте программы функцию `iprintLF` на `iprint` (рис. 4.11).



```

GNU nano 6.2 /home/sunnatillo/work/arch-pc/lab06/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

```

Рис. 4.11: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис. ??).

Вывод изменился, потому что символ переноса строки отображался когда программа исполнялась с функцией `iprintLF`, а `iprint` добавляет к выводу символ переноса строку в отличие от `iprintLF`.

```

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

```

Рис. 4.12: Запуск исполняемого файла

#Выполнение арифметических операций в NASM

Создаю файл `lab6-3.asm` в каталоге `~/work/arch-pc/lab06`: (рис. 4.13).

```

10sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$

```

Рис. 4.13: lab6-3.asm

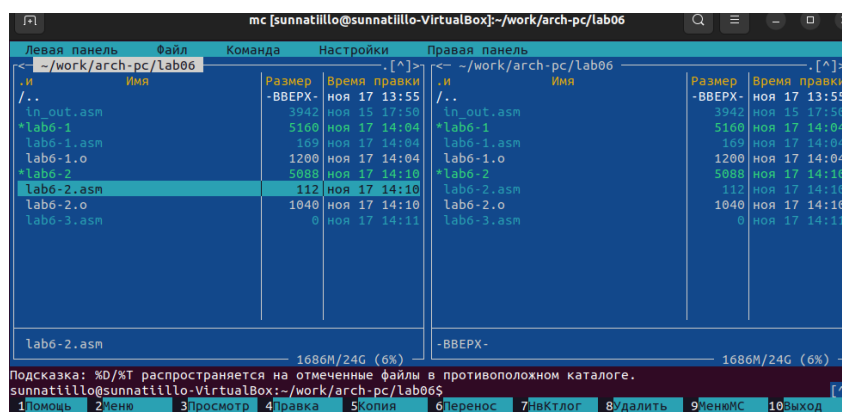


Рис. 4.14: lab6-3.asm

из листинга 6.3 и введите в lab6-3.asm.(рис. 4.21).

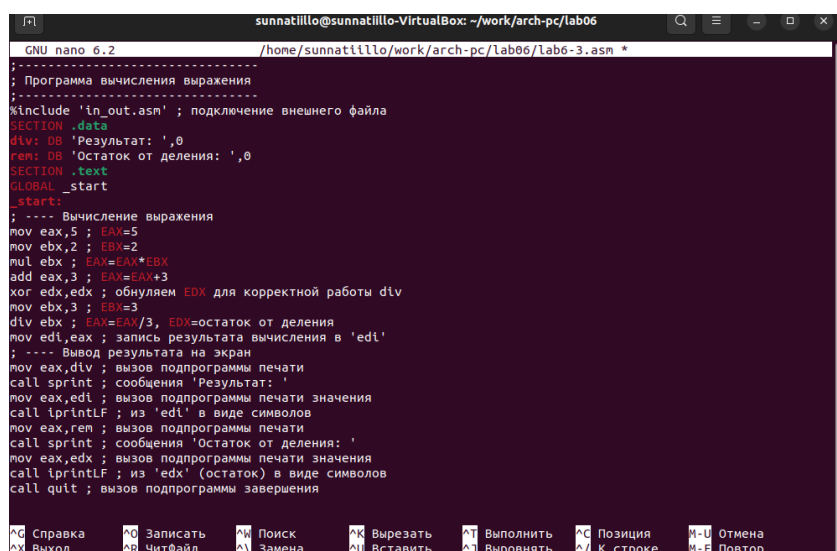


Рис. 4.15: Я ввел текст

Листинг 6.3. Программа вычисления выражения

Создаю исполняемый файл и запускаю его. Результат работы программы должен быть следующим:(рис. 4.21).

```

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

```

Рис. 4.16: Результат работы

Изменяю текст программы для вычисления выражения $(4 \times 6 + 2)/5$.
 .(рис. 4.21)

```

GNU nano 6.2 /home/sunnatillo/work/arch-pc/lab06/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,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 iprintf ; 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 iprintf ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

Рис. 4.17: Изменил текст

Создаю исполняемый файл и проверяю его работу (рис. 4.21)

```

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

```

Рис. 4.18: Проверил его работу

Создаю файл variant.asm в каталоге ~/work/arch-pc/lab06:(рис. 4.21) Листинг 6.4.
 Программа вычисления вычисления варианта задания по номеру студенческого билета

```

sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$

```

Рис. 4.19: Проверял его работу

Листинг 6.4. Программа вычисления варианта задания по номеру студенческого билета. (рис. 4.21) При выполнении данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM

```

sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$ nasm -f elf variant.asm
sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132224173
Ваш вариант: 14
sunnatillo@sunnatillo-VirtualBox:~/work/arch-pc/lab06$

```

Рис. 4.20: Проверял его работу

#Ответы на вопросы по программе

1. За вывод сообщения “Ваш вариант” отвечают строки кода:

```
mov eax,rem
```

```
call sprint
```

2. Инструкция mov еск, к используется, чтобы положить адрес вводимой строки в регистр есх mov edx, 80 запись в регистр ейх длины вводимой строки call sread вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры 2022-252
3. call atol используется для вызова подпрограммы из внешнего файла, которая преобразует всїї-код символа в целое число и записывает результат в регистр сах
4. За вычисления варианта отвечают строки:

ко ехех обнуление ех для корректной работы там «ок.20 сок - 20 div тах/20, dx остаток от деления ine edx edx estx 1

5. При выполнении инструкции `div ebx` остаток от деления записывается регистр `edx`
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1
7. За вывод на экран результатов вычислений отвечают строки:

```
mov eax,
eds call iprintLF
```

#Задание для самостоятельной работы

Написать программу вычисления выражения $x = x(x)$. 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 $x(x)$ 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.(рис. 4.21)


```

%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; секция инициализированных данных
msg: DB 'Введите значение переменной x: ',0
rem: DB 'Результат: ',0
SECTION .bss ; секция не инициализированных данных
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры, выделенный ре
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
; ---- Вычисление выражения
mov eax, msg ; запись адреса выводимого сообщения в eax
call sprintf ; вызов подпрограммы печати сообщения
mov ecx, x ; запись адреса переменной в ecx
mov edx, 80 ; запись длины вводимого значения в edx
call cread ; вызов подпрограммы ввода сообщения
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'
add eax,11; eax = eax+11 = x + 11
mov ebx,2 ; запись значения 2 в регистр ebx
mul ebx; EAX=EAX*EBX = (x+11)*2
add eax,-6; eax = eax-6 = (x+11)*2-6
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,rem ; вызов подпрограммы печати
call sprintf ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprint ; из 'edi' в виде символов
call quit ; вызов подпрограммы завершения

```

Рис. 4.21: Проверил его работу

Wininclude 'as * m' ; подключение внешнего файла
SECTION data; секция инициализированных данных
msg: DB 'Введите значение переменной x:',0
rem: DB 'Результат:',0
SECTION bss; секция не инициализированных данных
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры, выделенный ре
SECTION text; Код программы
GLOBAL _start; Начало программы
_start: Точка входа в программу
Вычисление выражения
mov eax, ns; запись адреса выводимого сообщения в сах

`call sprint`; вызов подпрограммы печати сообщения
`mov ecx, x`; запись адреса переменной в `ecx` `mov edx, 80`; запись длины вводимого значения в
`ecx`
`call sread`; вызов подпрограммы ввода сообщения
`mov eax, x`; вызов подпрограммы преобразования
`call atoi`; ASCII кода в число, `saхих`
`add eax, 11`; $eax = eax + 11 = x + 11$
`mov ebx, 2`; запись значения 2 в регистр `ecx`
`mul ebx`; $EAX = EAX * EBX = (x + 11) * 2$
`add eax, -6`;
 $eax = eax - 6 = (x + 11) * 2 - 6$
`mov edi, eax`; запись результата вычисления в `'edi'`
Вывод результата на экран
`mov eax, ret`, вызов подпрограммы печати
`call sprint`: сообщения Результат:
`mov eax, edi`; вызов подпрограммы печати значения `call iprint`; из `'edi'` в виде
СИМВОЛОВ
`call quit`, вызов подпрограммы завершения

5 Выводы

При выполнении данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM