

Лабораторная работа №6

НПИбд-01-25 №1032252598

Иванова Ангелина Олеговна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Задания лабораторной работы	6
2.2	Задание для самостоятельной работы	19
3	Выводы	22

Список иллюстраций

2.1	Создание каталога и файла	6
2.2	Измененный файл lab6-1.asm	7
2.3	Создание исполняемого файла и вывод его работы	8
2.4	Измененный текст программы	9
2.5	Создание исполняемого файла и вывод его работы	10
2.6	Создание файла lab6-2.asm	11
2.7	Создание исполняемого файла и вывод его работы	12
2.8	Измененный текст программы	13
2.9	Создание исполняемого файла и вывод его работы	14
2.10	Создание исполняемого файла и вывод его работы	15
2.11	Файл lab6-3.asm	16
2.12	Создание исполняемого файла с изменениями и его запуск	16
2.13	Создание исполняемого файла с изменениями и его запуск	18
2.14	Создание исполняемого файла с изменениями и его запуск	18
2.15	Работа созданного нами файла	21

Список таблиц

1 Цель работы

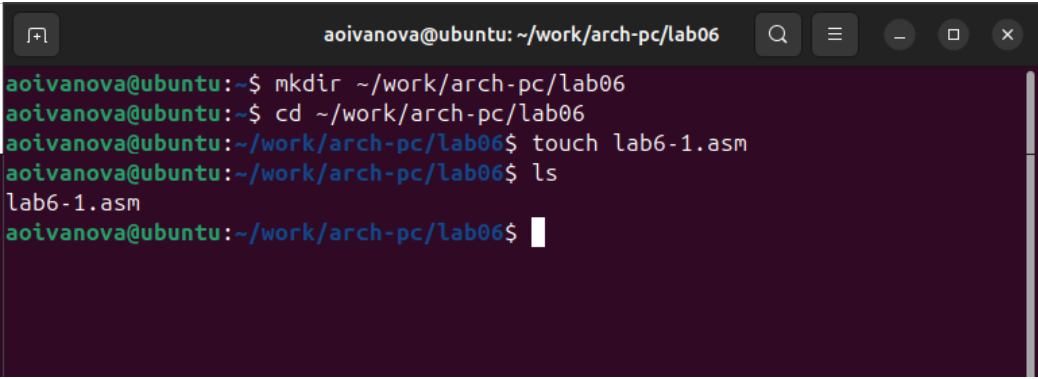
Целью данной лабораторной работы является освоение арифметических инструкций языка ассемблера NASM

2 Выполнение лабораторной работы

2.1 Задания лабораторной работы

2.1.1 Символьные и численные данные в NASM

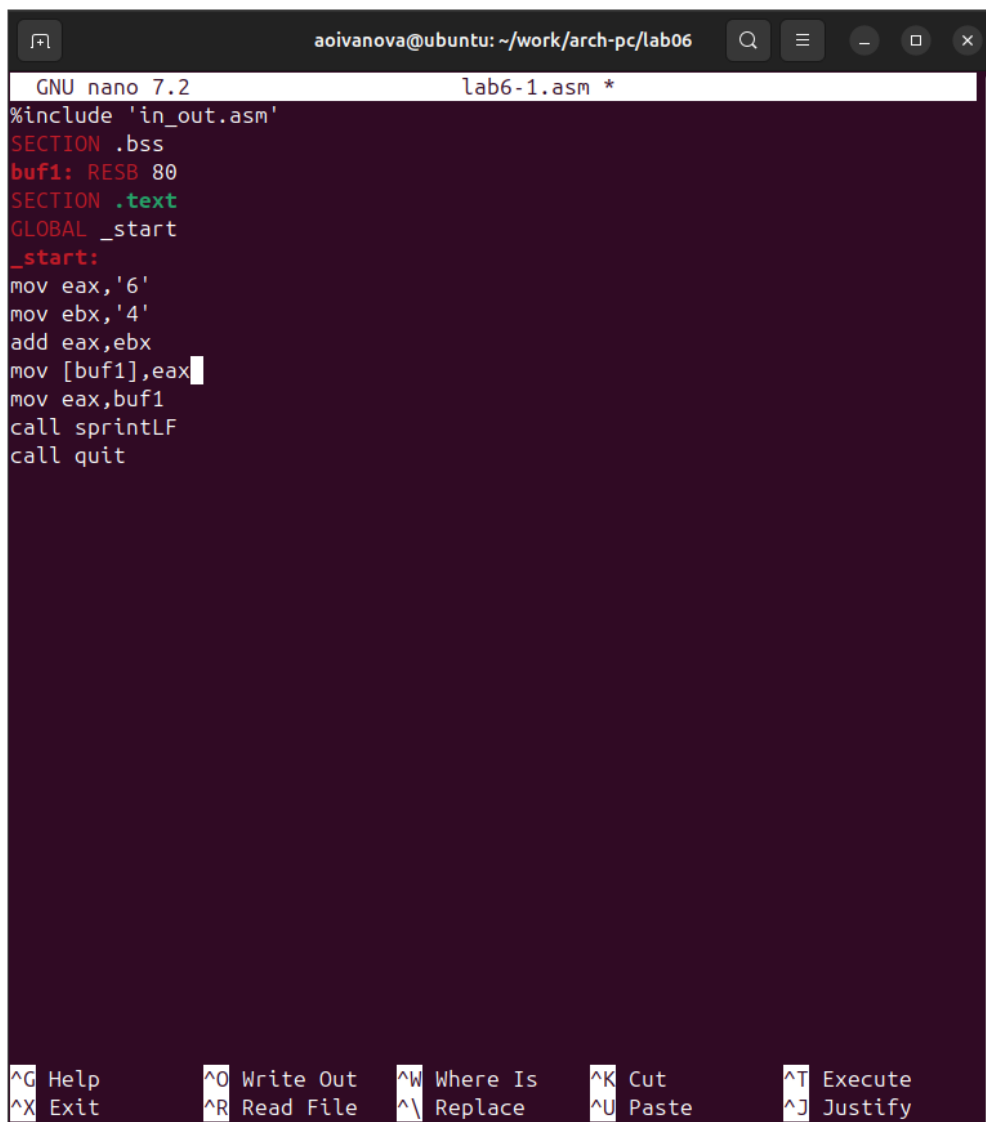
Создали каталог для программ лабораторной работы № 6, перейдите в него и создали файл lab6-1.asm



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

Рисунок 2.1: Создание каталога и файла

Приступили к рассмотрению примеров программ вывода символьных и численных значений. Ввели в файл lab6-1.asm текст программы из первого предоставленного нам листинга



```
GNU nano 7.2 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 sprintLF
call quit
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

Рисунок 2.2: Измененный файл lab6-1.asm

Создали исполняемый файл и запустили его. В данном случае при выводе значения регистра `eax` мы ожидали увидеть число 10. Однако результатом вывода является символ `j`. Это происходит потому, что код символа 6 равен 00110110 в двоичном представлении, а код символа 4 – 00110100. Команда `add eax,ebx` запишет в регистр `eax` сумму кодов – 01101010, что в свою очередь является кодом символа `j`

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

Рисунок 2.3: Создание исполняемого файла и вывод его работы

Далее изменили текст программы и вместо символов, записали в регистры числа.

Заменяли строки

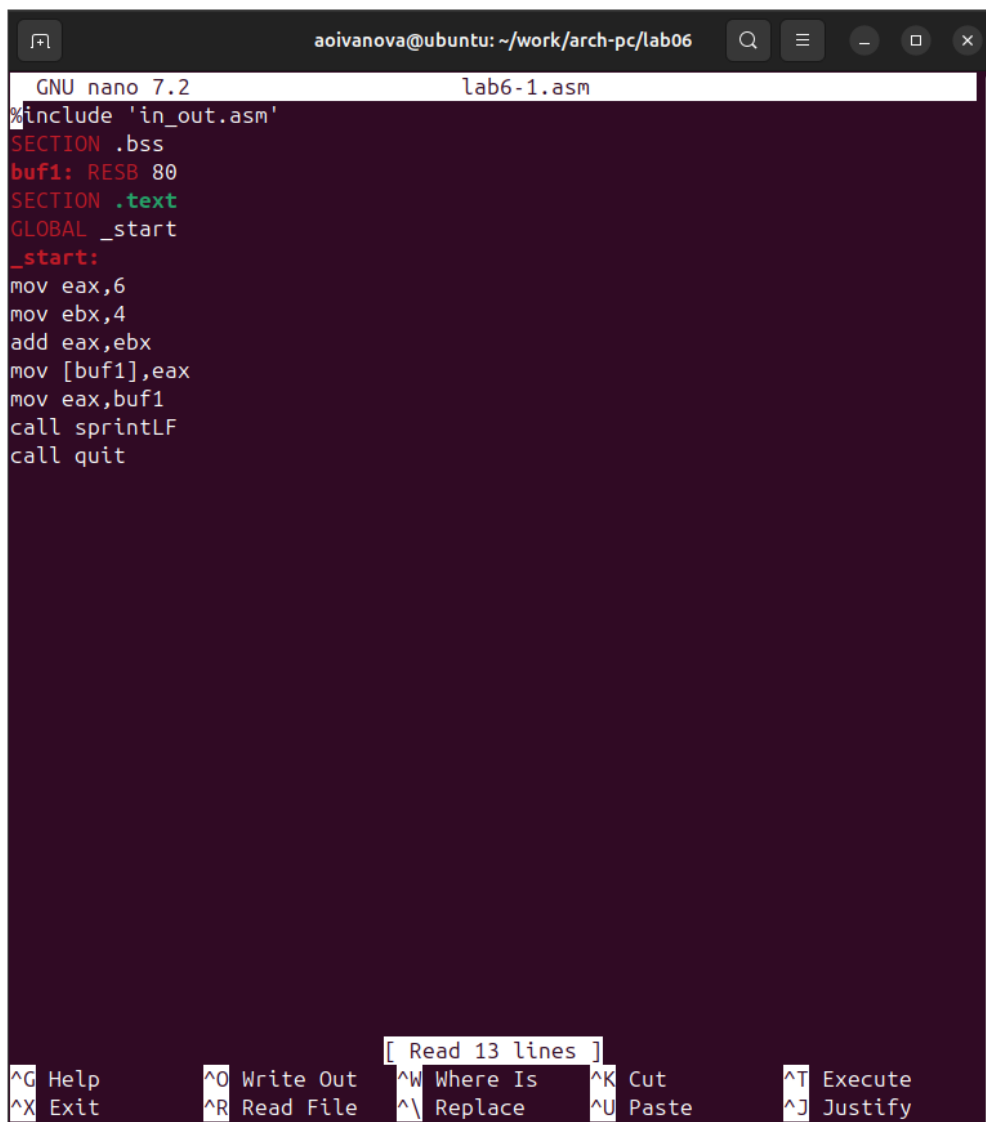
`mov eax,,6`

`mov ebx,,4`

на строки

`mov eax,6`

`mov ebx,4`



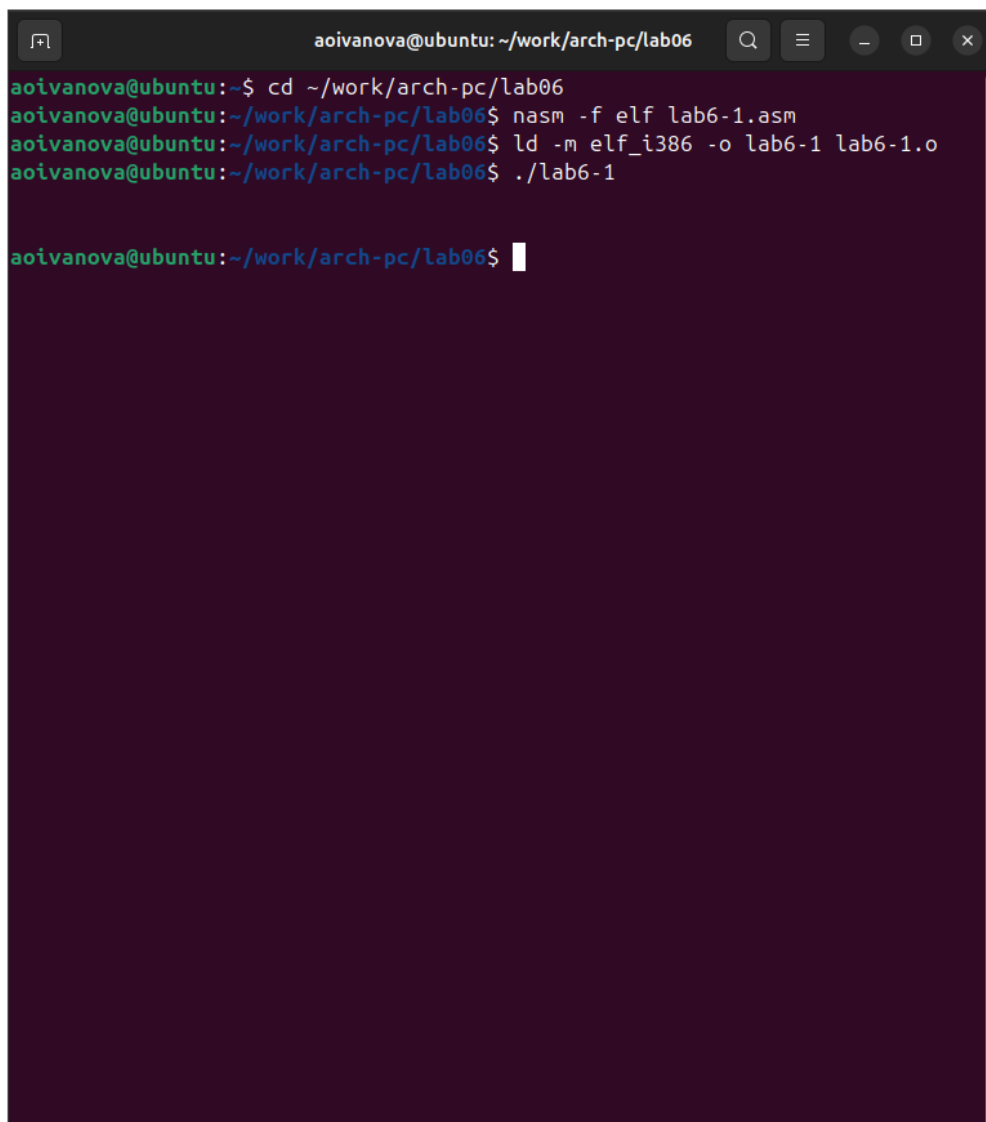
```
GNU nano 7.2 lab6-1.asm
#include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call printf
call _exit
```

[Read 13 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^I Replace	^U Paste	^J Justify

Рисунок 2.4: Измененный текст программы

Создали исполняемый файл и запустили его. Как и в предыдущем случае при исполнении программы мы не получим число 10. В данном случае выводится символ с кодом 10 - красная строка. Таким образом вывелось 2 пустые строки

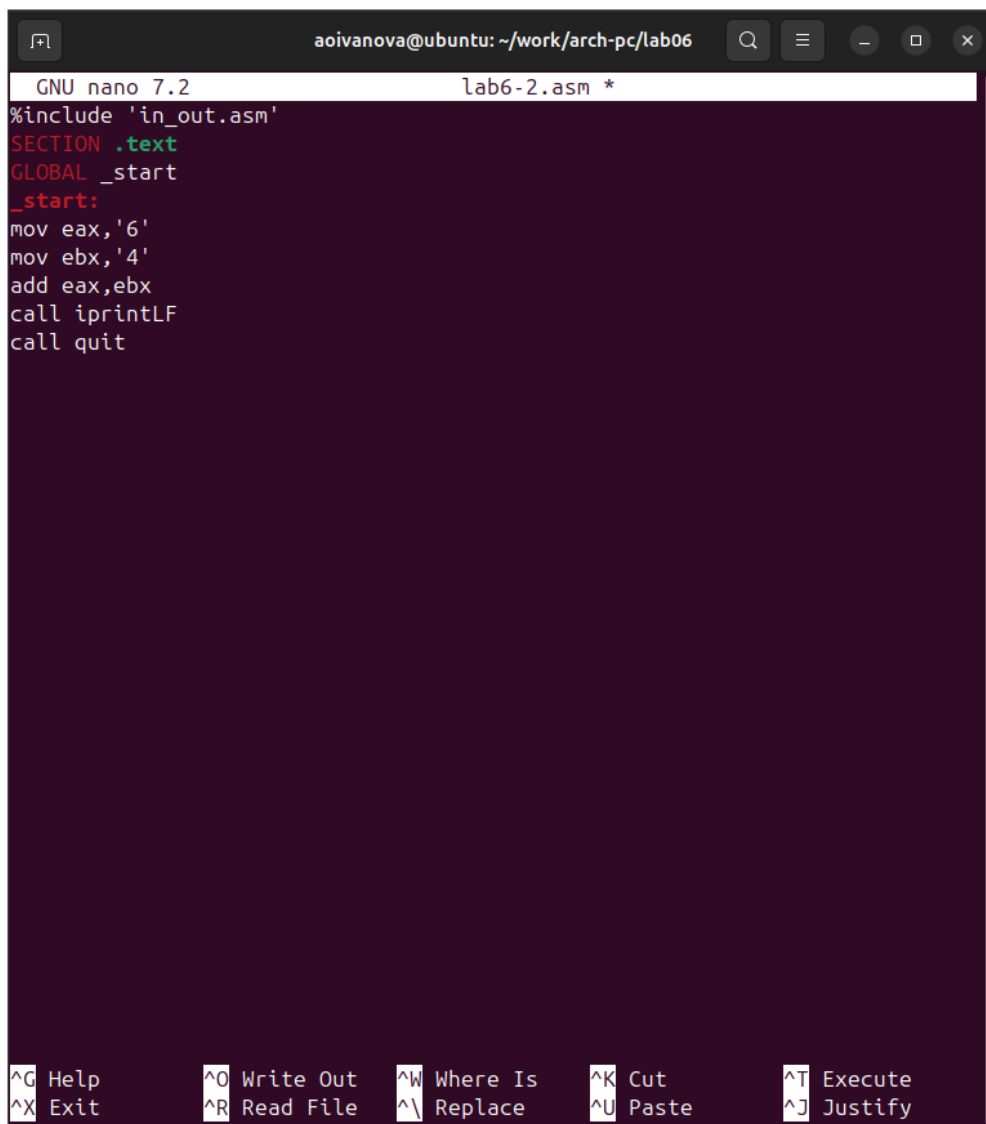
A terminal window with a dark purple background and light green text. The window title is 'aoivanova@ubuntu: ~/work/arch-pc/lab06'. The terminal shows the following commands and their outputs:

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

aoivanova@ubuntu:~/work/arch-pc/lab06$
```

Рисунок 2.5: Создание исполняемого файла и вывод его работы

Создали файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и ввели в него текст программы из второго листинга



```
GNU nano 7.2 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
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

Рисунок 2.6: Создание файла lab6-2.asm

Создали исполняемый файл и запустили его. В результате работы программы мы получили число 106. В данном случае, как и в первом, команда add складывает коды символов „6“ и „4“ ($54+52=106$). Однако, в отличие от программы из первого листинга, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число.

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

Рисунок 2.7: Создание исполняемого файла и вывод его работы

Аналогично предыдущему примеру изменили символы на числа. Заменяли строки `mov eax,,6`

`mov ebx,,4`

на строки

`mov eax,6`

`mov ebx,4`



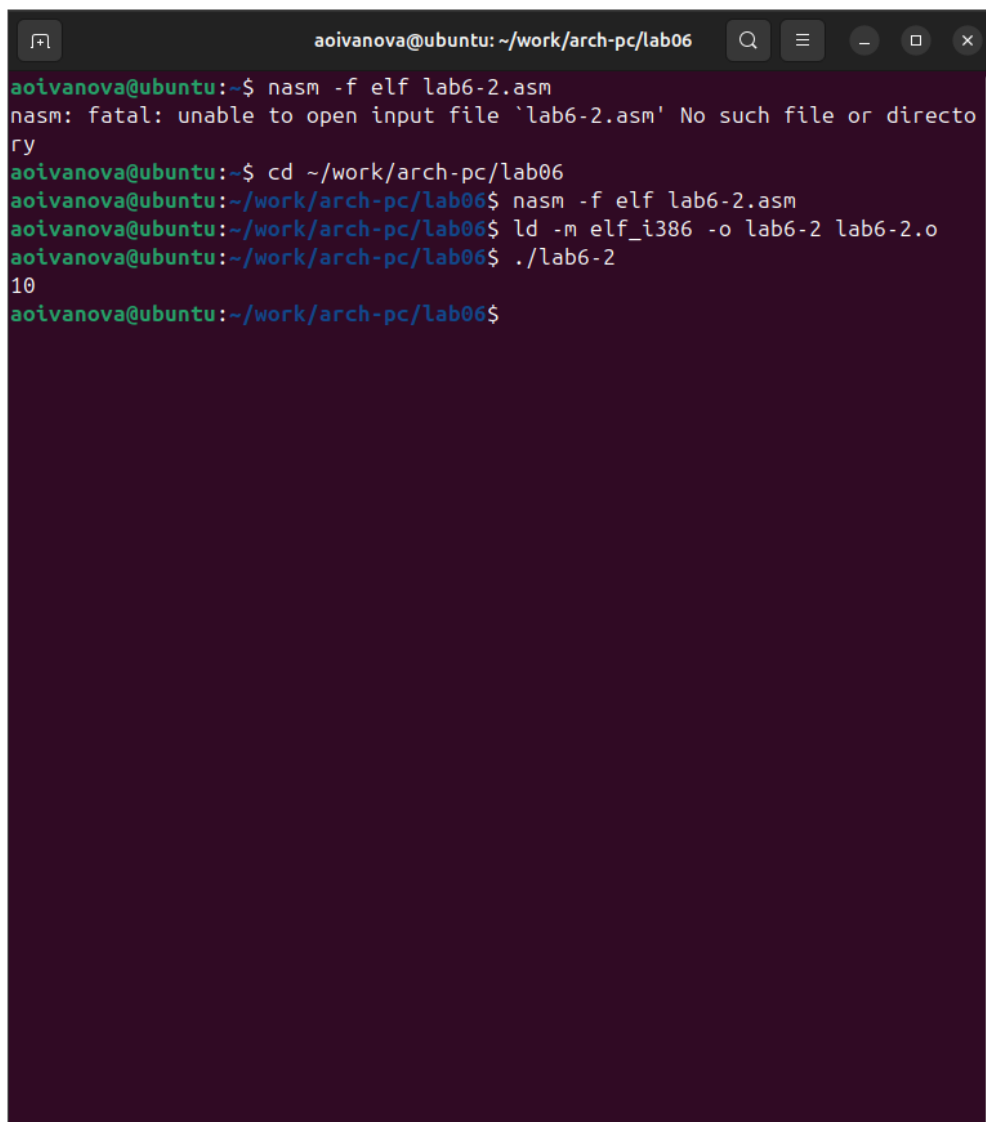
```
GNU nano 7.2 lab6-2.asm
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

[Wrote 9 lines]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute
^X Exit	^R Read File	^\ Replace	^U Paste	^J Justify

Рисунок 2.8: Измененный текст программы

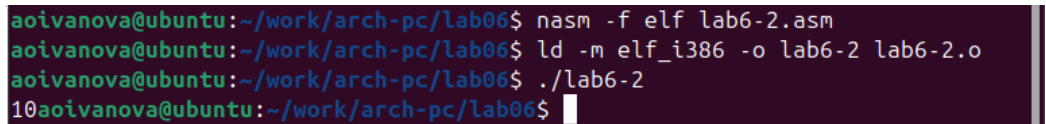
Создали исполняемый файл и запустили его. На вводе получили число 10.

A terminal window with a dark background and light green text. The window title is 'aoivanova@ubuntu: ~/work/arch-pc/lab06'. The terminal shows the following commands and output:

```
aoivanova@ubuntu:~$ nasm -f elf lab6-2.asm
nasm: fatal: unable to open input file `lab6-2.asm' No such file or directory
aoivanova@ubuntu:~$ cd ~/work/arch-pc/lab06
aoivanova@ubuntu:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
aoivanova@ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
aoivanova@ubuntu:~/work/arch-pc/lab06$ ./lab6-2
10
aoivanova@ubuntu:~/work/arch-pc/lab06$
```

Рисунок 2.9: Создание исполняемого файла и вывод его работы

Заменяли функцию `iprintlnf` на `iprintln`. Создали исполняемый файл и запустили его. Вывод функций `iprintlnf` и `iprintln` отличаются наличием и отсутствием перехода на новую строку. Использование функции `iprintlnf` содержит красную строку, а `iprintln` - не содержит.

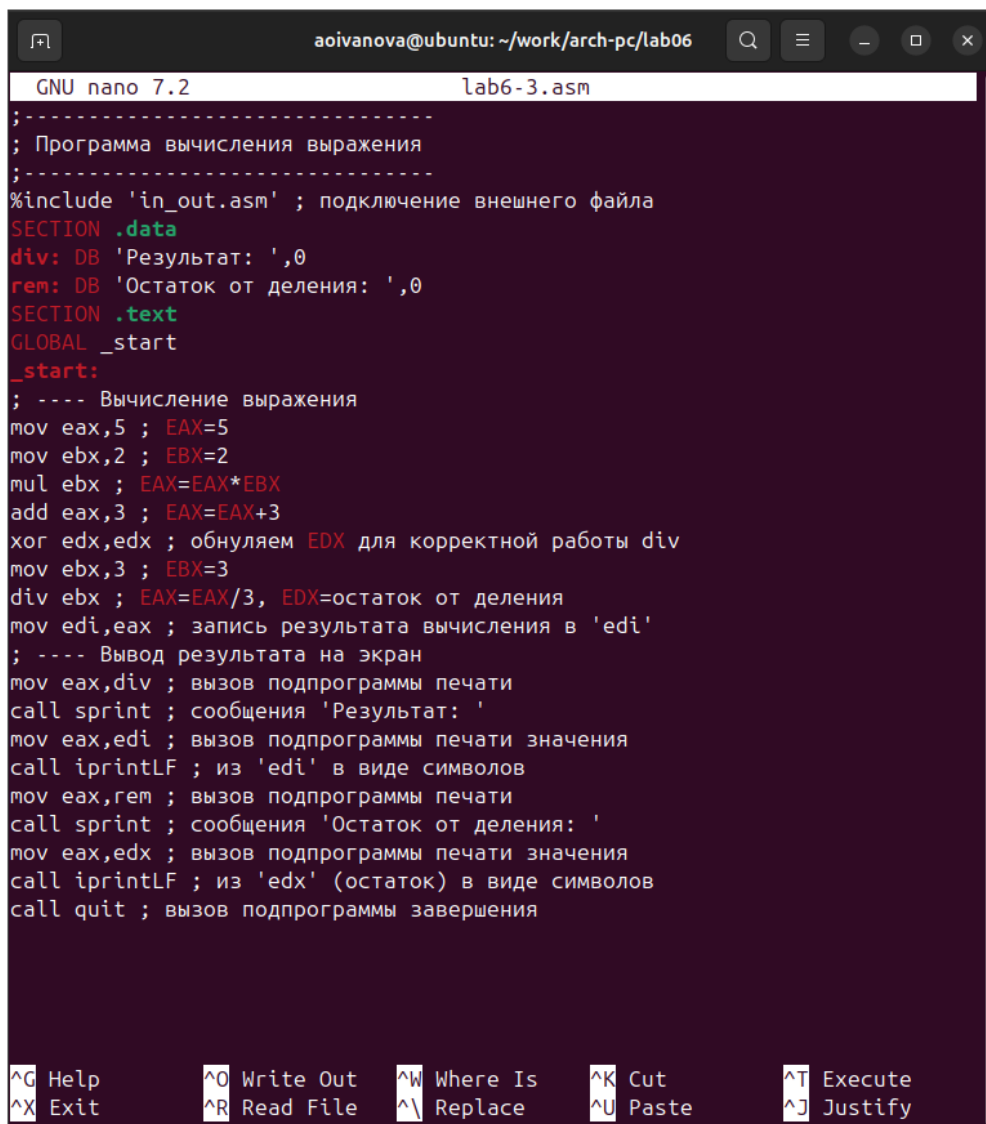
A terminal window with a dark background and light green text. It shows four lines of commands and their execution. The first line is 'nasm -f elf lab6-2.asm', the second is 'ld -m elf_i386 -o lab6-2 lab6-2.o', the third is './lab6-2', and the fourth is a prompt '10' followed by the same path and a cursor. The user is 'aoivanova' and the host is 'ubuntu'.

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

Рисунок 2.10: Создание исполняемого файла и вывод его работы

2.1.2 Выполнение арифметических операций в NASM

Создали файл lab6-3.asm в каталоге ~/work/arch-pc/lab06. Внимательно изучили текст программы из третьего листинга и ввели его в lab6-3.asm.

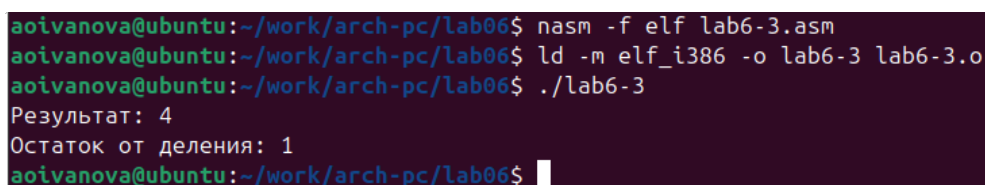


```
GNU nano 7.2 lab6-3.asm
;-----
; Программа вычисления выражения
;-----
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul 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

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```

Рисунок 2.11: Файл lab6-3.asm

Создали исполняемый файл и запустили его.



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

Рисунок 2.12: Создание исполняемого файла с изменениями и его запуск

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

Листинг 1

```
;-----  
; 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

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

Рисунок 2.13: Создание исполняемого файла с изменениями и его запуск

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: $(S_n \bmod 20) + 1$, где S_n – номер студенческого билета (В данном случае $a \bmod b$ – это остаток от деления a на b).
- вывести на экран номер варианта

Создали файл variant.asm в каталоге ~/work/arch-pc/lab06. Внимательно изучили текст программы из четвертого листинга и ввели в файл variant.asm Создайли исполняемый файл и запустили его

```
aoivanova@ubuntu:~/work/arch-pc/lab06$ nasm -f elf variant.asm
aoivanova@ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
aoivanova@ubuntu:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032252598
Ваш вариант: 19
```

Рисунок 2.14: Создание исполняемого файла с изменениями и его запуск

1. Строки, отвечающие за вывод сообщения „Ваш вариант:“:

```
mov eax,rem
```

```
call sprint
```

Где rem определено как:

```
rem: DB „Ваш вариант:“,0
```

2. Для чего используются следующие инструкции?
`mov ecx, x` ; помещает адрес буфера x в ecx
`mov edx, 80` ; устанавливает размер буфера
`call sread` ; вызывает функцию чтения ввода с клавиатуры
3. Функция `atoi` преобразует строку (введенный номер студенческого) в целое число. Результат сохраняется в регистре `eax`
4. Строки, отвечающие за вычисления варианта: `xor edx,edx` ; `edx=0`
`mov ebx,20` ; `ebx = 20`
`div ebx` ; `eax/ebx`
`inc edx` ; увеличение остатка на 1
5. При выполнении инструкции `div ebx` остаток от деления записывается в регистр `edx`.
6. Инструкция увеличивает значение в регистре `EDX` на 1, чтобы варианты нумеровались с 1 до 20, а не с 0 до 19.
7. Строки, отвечающие за вывод результата: `mov eax,rem` ; загрузка адреса строки «Ваш вариант:» `call sprint` ; вывод строки `mov eax,edx` ; загрузка номера варианта `call iprintLF` ; вывод числа с переводом строки

2.2 Задание для самостоятельной работы

Написали программу вычисления выражения $y = f(x)$. Программа выводит выражение для вычисления, выводит запрос на ввод значения x , вычисляет заданное выражение в зависимости от введенного x , выводит результат вычислений. Вид функции $f(x) = (1/3x + 5) * 7$. Создали исполняемый файл и проверили его работу для значений x_1 и x_2 .

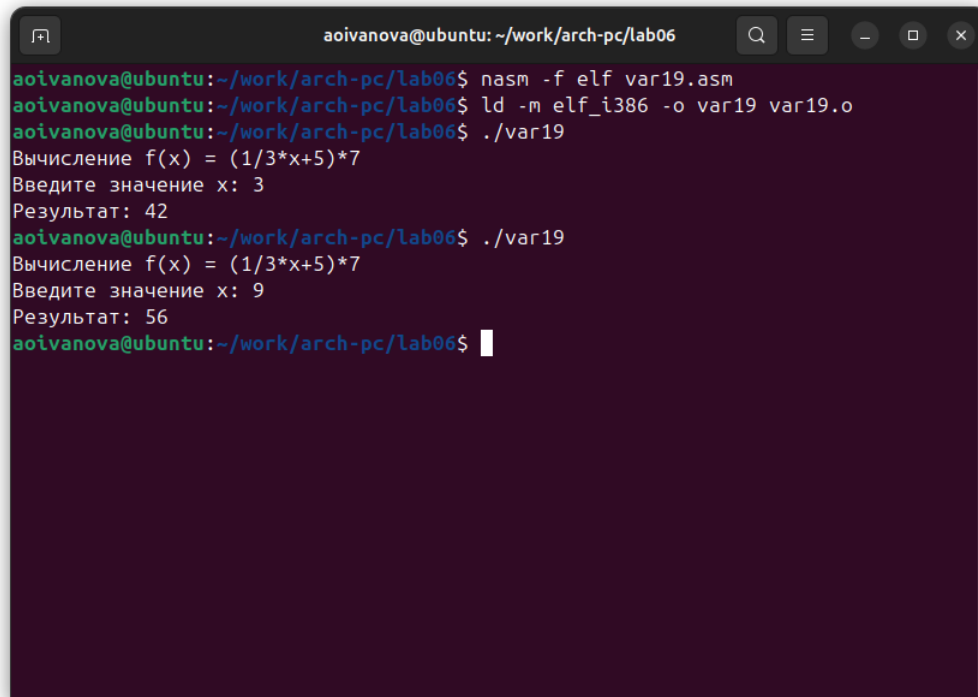
Листинг 2

```
%include „in_out.asm“ SECTION .data
msg: DB „Введите значение x:“,0
msgg: DB „Вычисление f(x) = (1/3x+5)7“,0
```

```

div: DB „Резултат:“,0
    SECTION .bss
x: RESB 80
    SECTION .text
GLOBAL _start
_start:
    mov eax, msgg
call sprintLF
    mov eax, msg
call sprint
    mov ecx, x
mov edx, 80
call sread
    mov eax, x
call atoi
    mov ebx, 3
xor edx, edx
div ebx
add eax, 5
mov ebx, 7
mul ebx
mov edi, eax
    mov eax, div
call sprint
mov eax, edi
call iprintLF
    call quit

```

A terminal window with a dark background and light-colored text. The window title is 'aoivanova@ubuntu: ~/work/arch-pc/lab06'. The terminal shows the following commands and output:

```
aoivanova@ubuntu:~/work/arch-pc/lab06$ nasm -f elf var19.asm
aoivanova@ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 -o var19 var19.o
aoivanova@ubuntu:~/work/arch-pc/lab06$ ./var19
Вычисление f(x) = (1/3*x+5)*7
Введите значение x: 3
Результат: 42
aoivanova@ubuntu:~/work/arch-pc/lab06$ ./var19
Вычисление f(x) = (1/3*x+5)*7
Введите значение x: 9
Результат: 56
aoivanova@ubuntu:~/work/arch-pc/lab06$
```

Рисунок 2.15: Работа созданного нами файла

Как можно заметить, программа корректно считает результат при введении x_1 , x_2

3 Выводы

Освоили арифметические инструкции языка ассемблера NASM