

Отчёта по лабораторной работе 6

Освоение арифметических инструкций языка ассемблера NASM.

Субанов Юсуф Жура угли НПМбв-01-21

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	19

Список иллюстраций

2.1	Пример программы	7
2.2	Работа программы	7
2.3	Пример программы	8
2.4	Работа программы	8
2.5	Пример программы	9
2.6	Работа программы	9
2.7	Пример программы	10
2.8	Работа программы	10
2.9	Работа программы	10
2.10	Пример программы	11
2.11	Работа программы	12
2.12	Пример программы	13
2.13	Работа программы	13
2.14	Пример программы	14
2.15	Работа программы	15
2.16	Пример программы	17
2.17	Работа программы	18

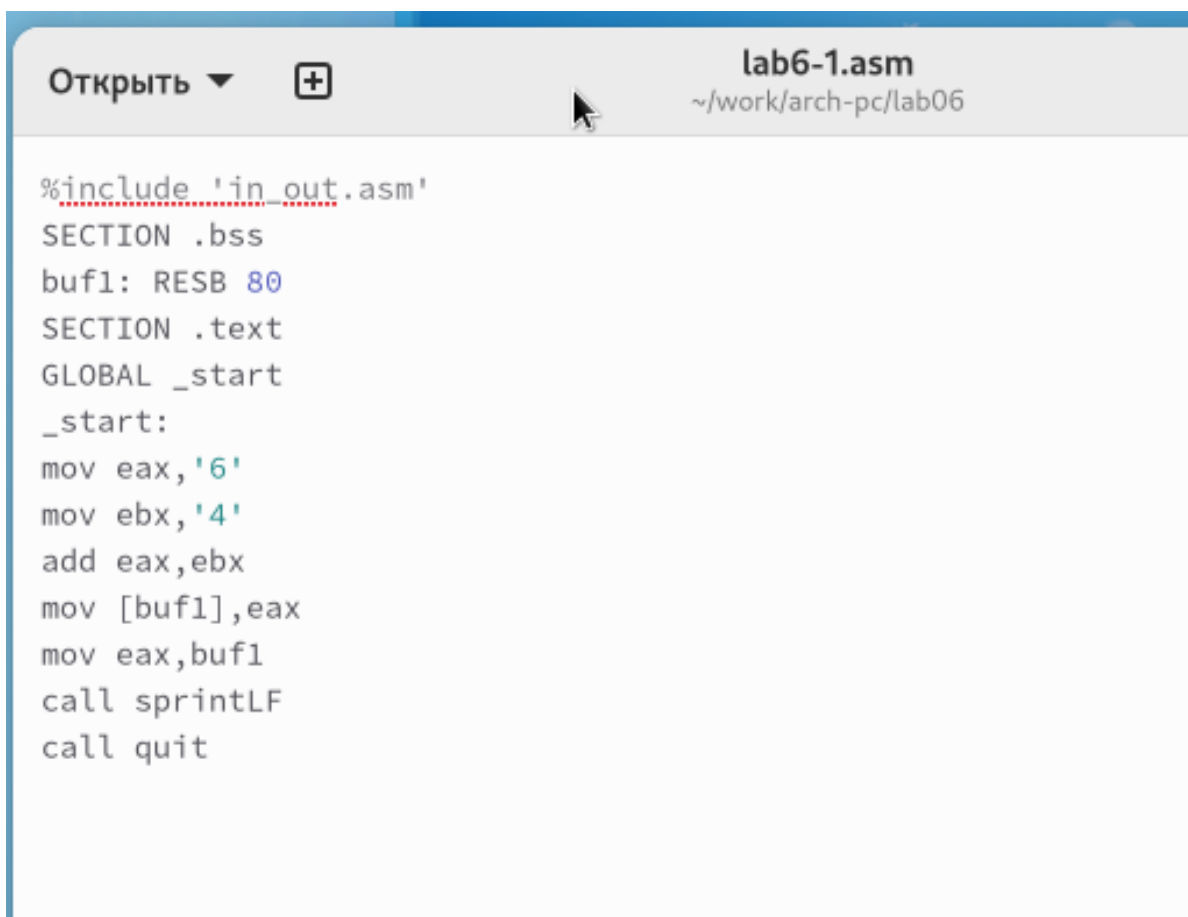
Список таблиц

1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

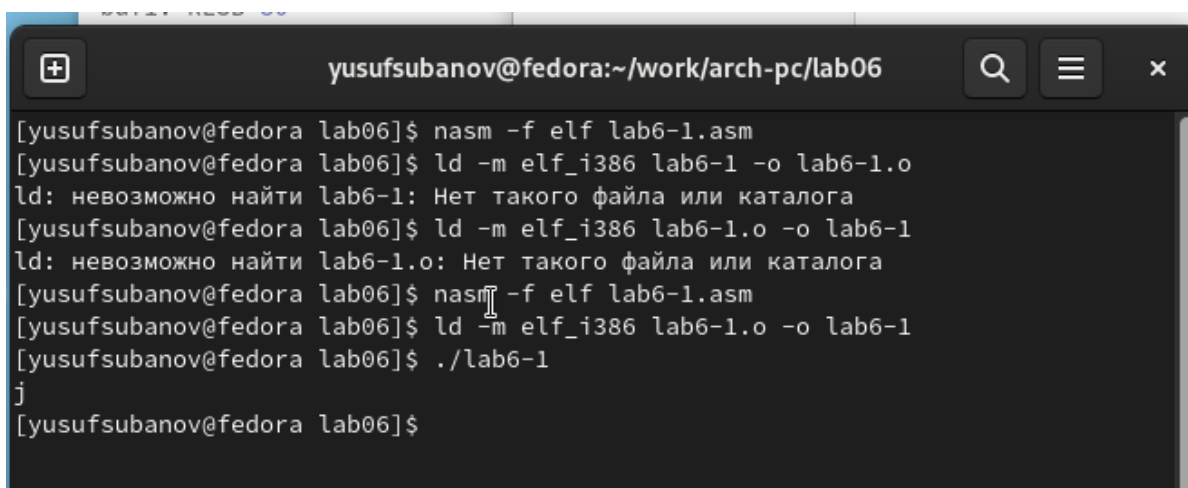
1. Создаем каталог для программ лабораторной работы № 6, переходим в него и создаем файл lab6-1.asm:
2. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистр eax.



```
Открыть ▾ + lab6-1.asm ~/work/arch-pc/lab06

%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. 2.1: Пример программы



```
yusufsubanov@fedora:~/work/arch-pc/lab06

[yusufsubanov@fedora lab06]$ nasm -f elf lab6-1.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-1 -o lab6-1.o
ld: невозможно найти lab6-1: Нет такого файла или каталога
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-1.o -o lab6-1
ld: невозможно найти lab6-1.o: Нет такого файла или каталога
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-1.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-1.o -o lab6-1
[yusufsubanov@fedora lab06]$ ./lab6-1
j
[yusufsubanov@fedora lab06]$
```

Рис. 2.2: Работа программы

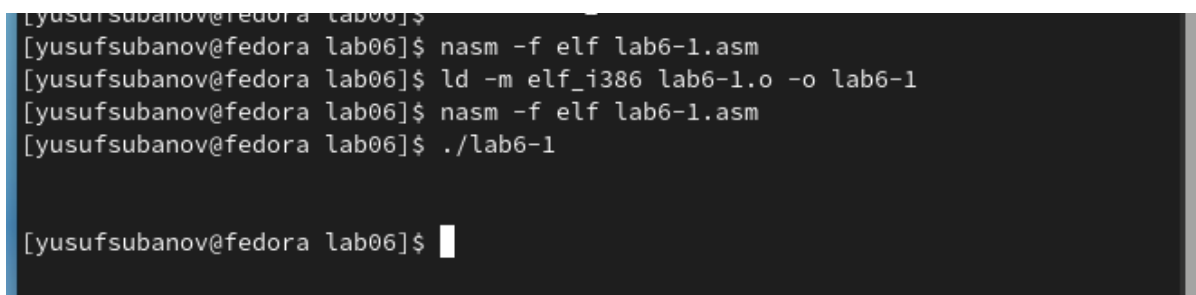
3. Далее изменим текст программы и вместо символов, запишем в регистры числа.



```
Открыть ▾ + lab6-1.asm
~/work/arch-pc/lab06

%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. 2.3: Пример программы



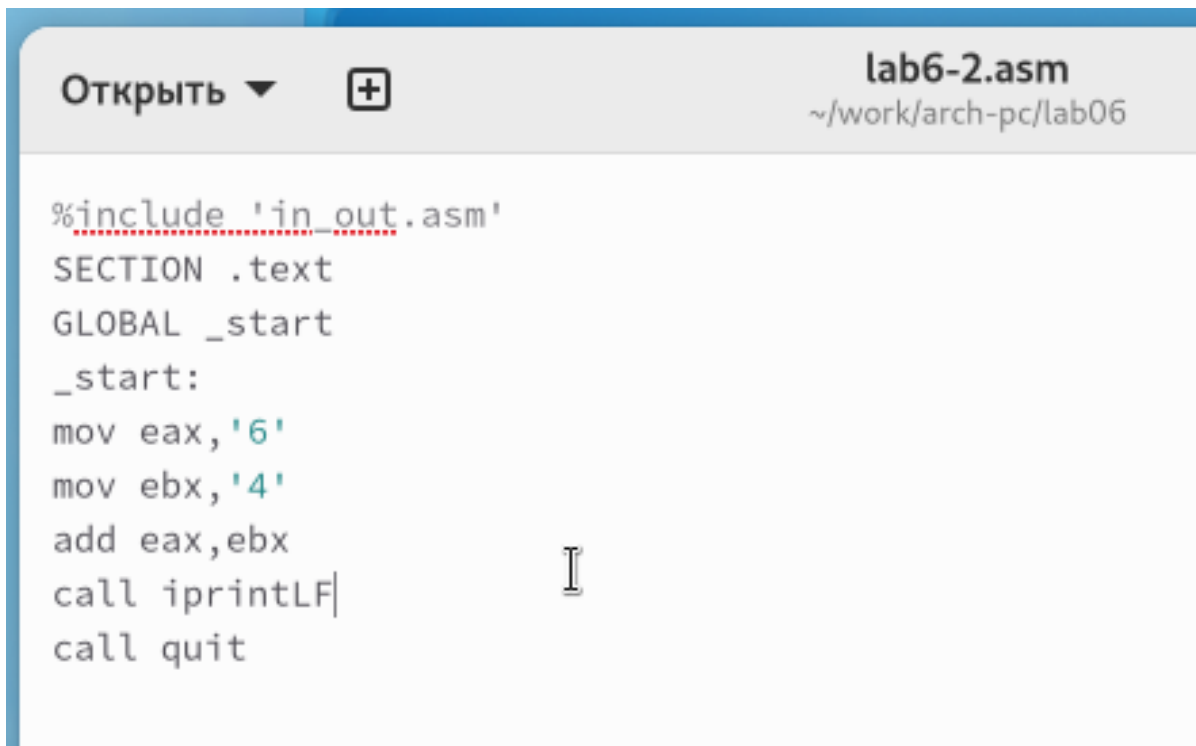
```
[yusufsubanov@fedora lab06]$
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-1.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-1.o -o lab6-1
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-1.asm
[yusufsubanov@fedora lab06]$ ./lab6-1

[yusufsubanov@fedora lab06]$
```

Рис. 2.4: Работа программы

Никакой символ не виден, но он есть. Это возврат каретки LF.

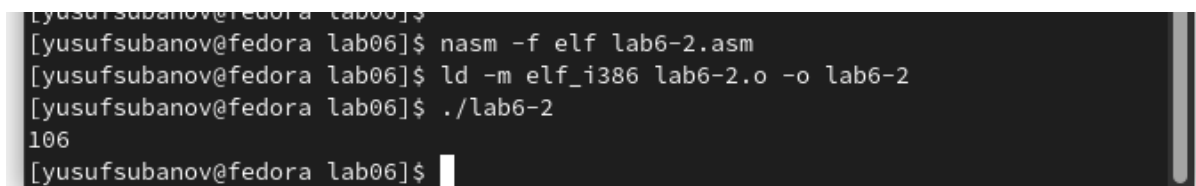
4. Как отмечалось выше, для работы с числами в файле in_out.asm реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы с использованием этих функций.



```
Открыть ▾ + lab6-2.asm
~/work/arch-pc/lab06

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

Рис. 2.5: Пример программы



```
[yusufsubanov@fedora lab06]$
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-2.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[yusufsubanov@fedora lab06]$ ./lab6-2
106
[yusufsubanov@fedora lab06]$
```

Рис. 2.6: Работа программы

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 отличие от прошлой программы, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число.

5. Аналогично предыдущему примеру изменим символы на числа.

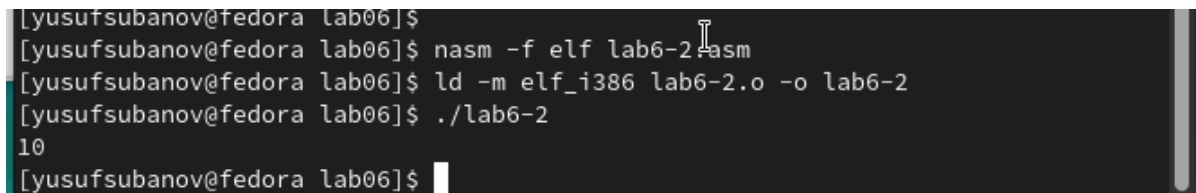
Создайте исполняемый файл и запустите его. Какой результат будет получен при исполнении программы? – получили число 10



```
Открыть ▾ + lab6-2.asm
~/work/arch-pc/lab06

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

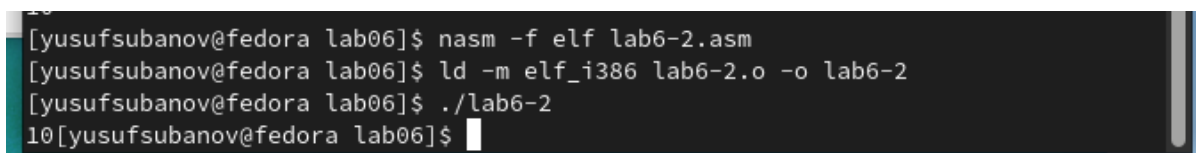
Рис. 2.7: Пример программы



```
[yusufsubanov@fedora lab06]$
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-2.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[yusufsubanov@fedora lab06]$ ./lab6-2
10
[yusufsubanov@fedora lab06]$
```

Рис. 2.8: Работа программы

Замените функцию `iprintLF` на `iprint`. Создайте исполняемый файл и запустите его. Чем отличается вывод функций `iprintLF` и `iprint`? - Вывод отличается что нет переноса строки. (рис. [2.9])



```
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-2.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[yusufsubanov@fedora lab06]$ ./lab6-2
10[yusufsubanov@fedora lab06]$
```

Рис. 2.9: Работа программы

6. В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения

$$f(x) = (5 * 2 + 3) / 3$$



```
lab6-3.asm
~/work/arch-pc/lab06

%include 'in_out.asm'
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:

mov eax,5
mov ebx,2
mul ebx
add eax,3
xor edx,edx
mov ebx,3
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
```

Рис. 2.10: Пример программы

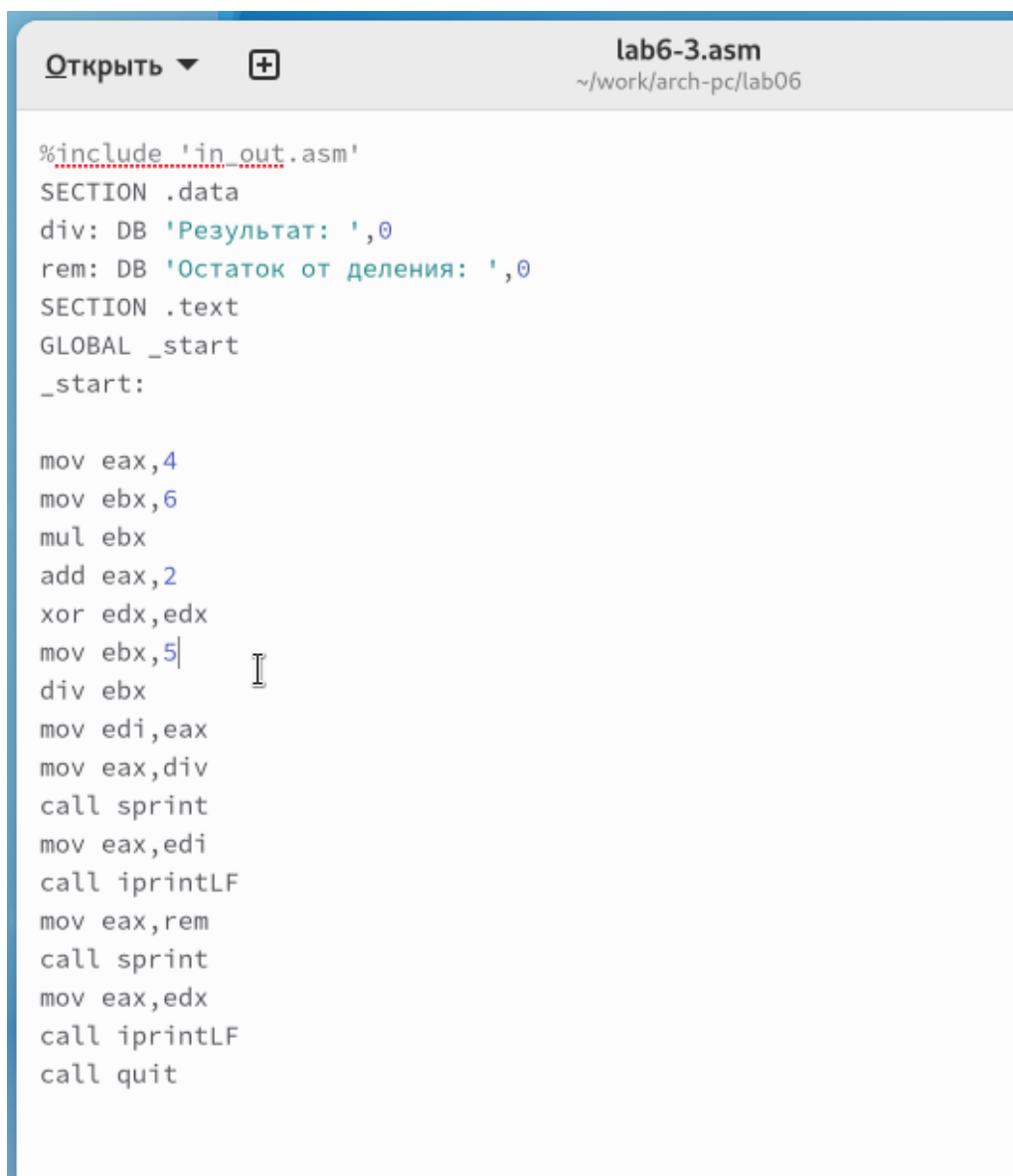
```
[yusufsubanov@fedora lab06]$  
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-3.asm  
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-3.o -o lab6-3  
[yusufsubanov@fedora lab06]$ ./lab6-3  
Результат: 4  
Остаток от деления: 1  
[yusufsubanov@fedora lab06]$
```

Рис. 2.11: Работа программы

Измените текст программы для вычисления выражения

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

. Создайте исполняемый файл и проверьте его работу.

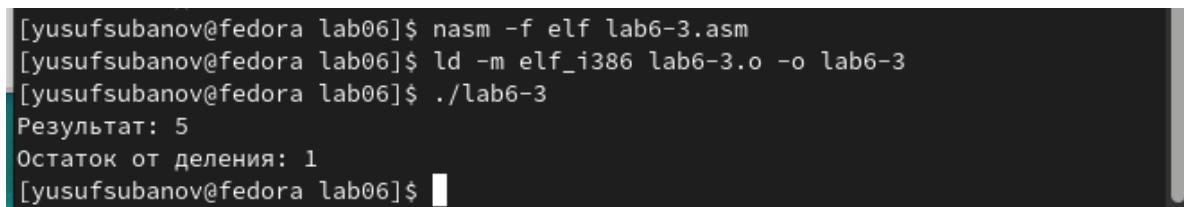


```
Открыть ▾ + lab6-3.asm
~/work/arch-pc/lab06

%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
```

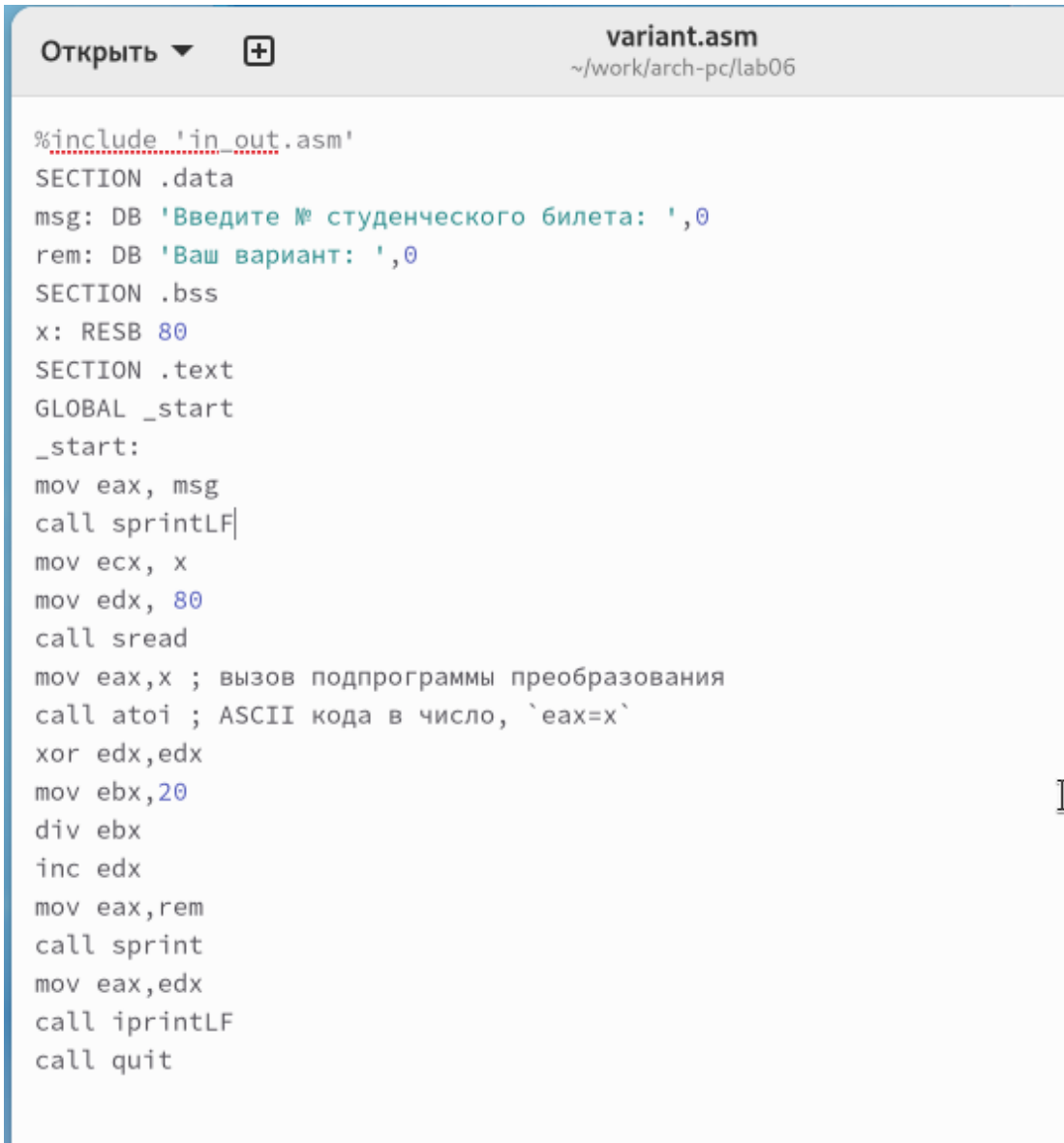
Рис. 2.12: Пример программы



```
[yusufsubanov@fedora lab06]$ nasm -f elf lab6-3.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 lab6-3.o -o lab6-3
[yusufsubanov@fedora lab06]$ ./lab6-3
Результат: 5
Остаток от деления: 1
[yusufsubanov@fedora lab06]$
```

Рис. 2.13: Работа программы

7. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета:



The screenshot shows a code editor window titled 'variant.asm' with the path '~/.work/arch-pc/lab06'. The code is written in assembly language and includes comments in Russian. It defines data and bss sections, then implements a program that prompts the user for a student ticket number, reads it, and calculates a variant number based on the ticket number modulo 20.

```
%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 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
inc edx
mov eax, rem
call sprint
mov eax, edx
call iprintLF
call quit
```

Рис. 2.14: Пример программы

```
[yusufsubanov@fedora lab06]$  
[yusufsubanov@fedora lab06]$  
[yusufsubanov@fedora lab06]$ nasm -f elf variant.asm  
[yusufsubanov@fedora lab06]$ ld -m elf_i386 variant.o -o variant  
[yusufsubanov@fedora lab06]$ ./variant  
Введите № студенческого билета:  
1032214640  
Ваш вариант: 1  
[yusufsubanov@fedora lab06]$
```

Рис. 2.15: Работа программы

- Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? – `mov eax,rem` – перекладывает в регистр значение переменной с фразой ‘Ваш вариант:’ `call sprint` – вызов подпрограммы вывода строки
- Для чего используются следующие инструкции? `nasm mov ecx, x mov edx, 80 call sread`

Считывает значение студбилета в переменную X из консоли

- Для чего используется инструкция “`call atoi`”? - эта подпрограмма переводит введенные символы в числовой формат
- Какие строки листинга 6.4 отвечают за вычисления варианта?

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

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

1 байт AH 2 байта DX 4 байта EDX – наш случай

- Для чего используется инструкция “`inc edx`”? по формуле вычисления варианта нужно прибавить единицу
- Какие строки листинга 6.4 отвечают за вывод на экран результата вычисления

mov eax,edx – результат перекладывается в регистр eax call iprintLF – вызов подпрограммы вывода

8. 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.

Получили вариант 1 -

$$(10 + 2x)/3$$

для $x=1$ и 10


```
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, 2
mul ebx
add eax, 10
mov ebx, 3
div ebx
|
mov ebx, eax
mov eax, rem
call sprintf
mov eax, ebx
call iprintLF
call quit
```

Рис. 2.16: Пример программы

```
[yusufsubanov@fedora lab06]$ nasm -f elf calc.asm
[yusufsubanov@fedora lab06]$ ld -m elf_i386 calc.o -o calc
[yusufsubanov@fedora lab06]$ ./calc
Введите X
1
выражение = : 4
[yusufsubanov@fedora lab06]$ ./calc
Введите X
10
выражение = : 10
[yusufsubanov@fedora lab06]$
```

Рис. 2.17: Работа программы

3 Выводы

Изучили работу с арифметическими операциями