

Отчёта по лабораторной работе 7

Освоение арифметических инструкций языка ассемблера NASM.

Чесноков Артемий Павлович НПИбд-02-22

Содержание

1	Выполнение лабораторной работы	5
2	Выводы	19

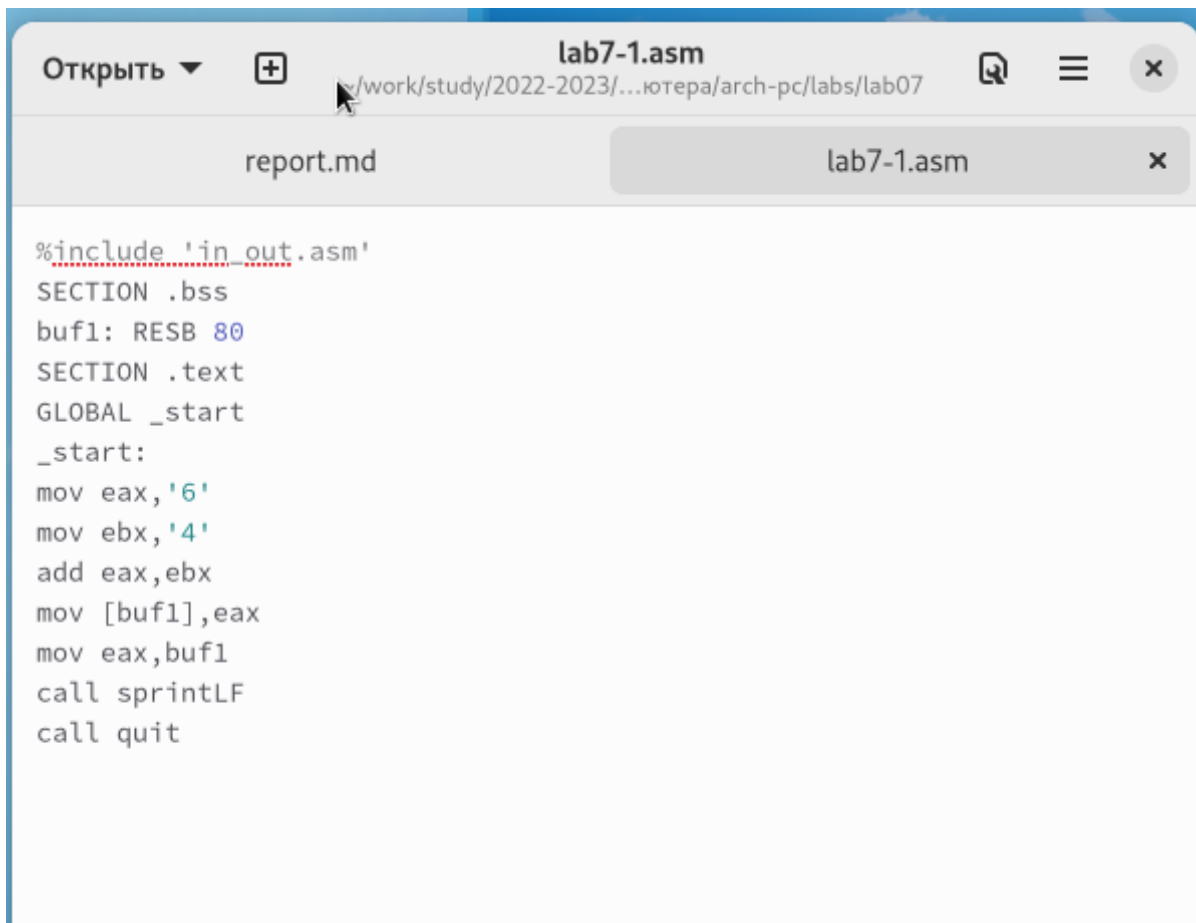
Список иллюстраций

1.1	Пример программы	6
1.2	Работа программы	6
1.3	Пример программы	7
1.4	Работа программы	8
1.5	Пример программы	8
1.6	Работа программы	9
1.7	Пример программы	10
1.8	Работа программы	10
1.9	Работа программы	11
1.10	Пример программы	12
1.11	Работа программы	12
1.12	Пример программы	13
1.13	Работа программы	14
1.14	Пример программы	15
1.15	Работа программы	15
1.16	Пример программы	17
1.17	Работа программы	18

Список таблиц

1 Выполнение лабораторной работы

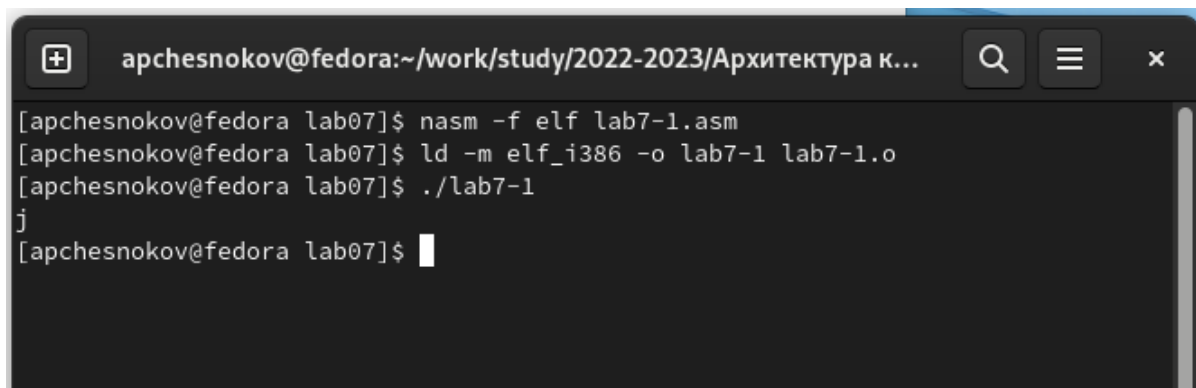
1. Создадим каталог для программ лабораторной работы № 6, перейдите в него и создадим файл lab7-1.asm:
2. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистр eax. (рис. 1.1, 1.2)



```
Открыть ▾ + lab7-1.asm
~/work/study/2022-2023/...ютеpa/arch-pc/labs/lab07
report.md lab7-1.asm x

%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. 1.1: Пример программы

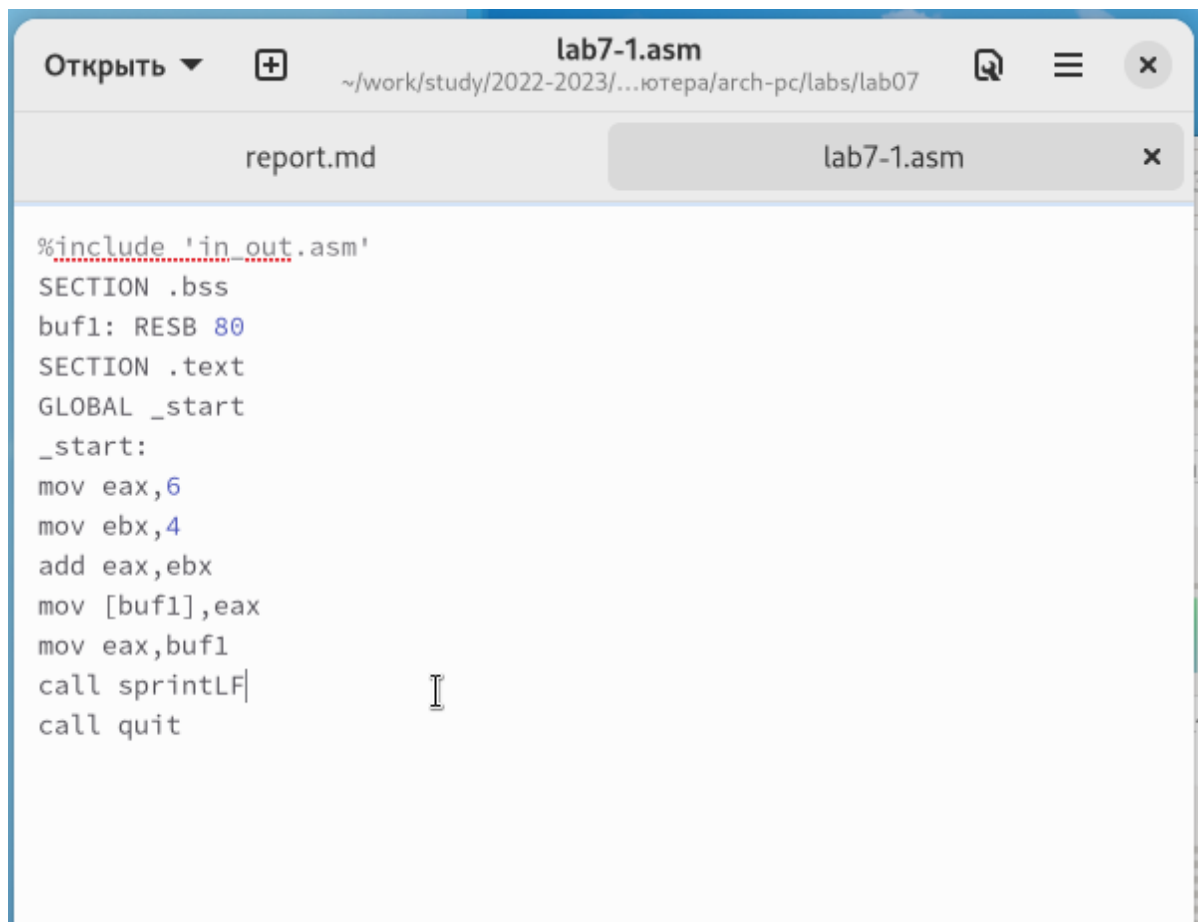


```
+ apchesnokov@fedora:~/work/study/2022-2023/Архитектура к... Q ≡ x
[apchesnokov@fedora lab07]$ nasm -f elf lab7-1.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[apchesnokov@fedora lab07]$ ./lab7-1
j
[apchesnokov@fedora lab07]$
```

Рис. 1.2: Работа программы

3. Далее изменим текст программы и вместо символов, запишем в регистры числа. Исправьте текст программы (Листинг 1) следующим образом: (рис.

1.3, 1.4)

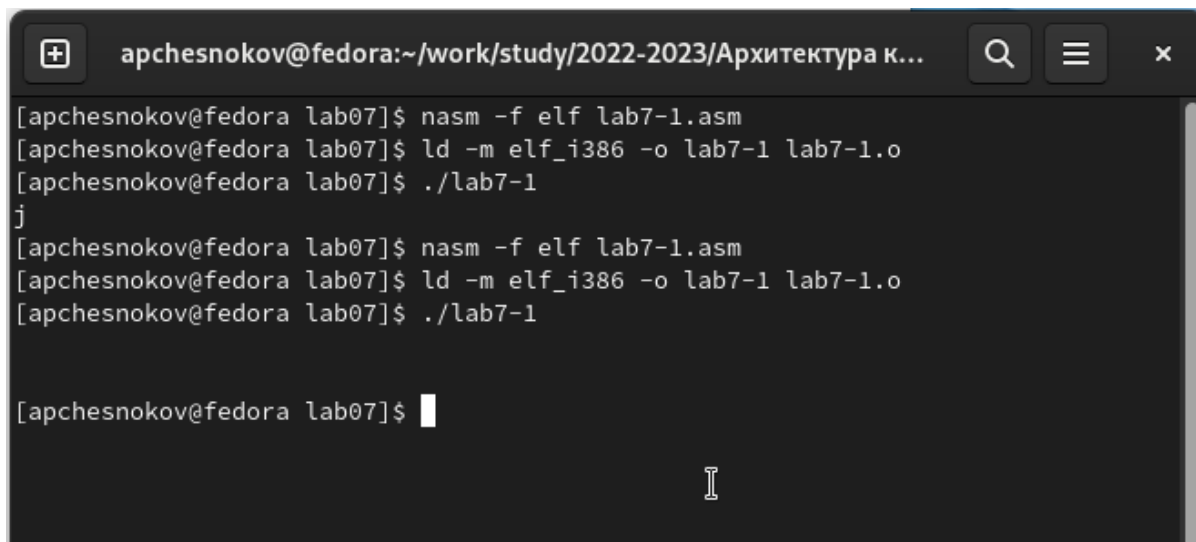


The image shows a code editor window with a tab titled 'lab7-1.asm'. The address bar shows the file path: '~/.work/study/2022-2023/...ютера/arch-pc/labs/lab07'. The editor contains the following assembly code:

```
%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
```

A cursor is visible on the line 'call sprintLF'.

Рис. 1.3: Пример программы

A terminal window with a dark background. The title bar shows the user 'apchesnokov@fedora' and the current directory '~/work/study/2022-2023/Архитектура к...'. The terminal contains the following commands and output:

```
[apchesnokov@fedora lab07]$ nasm -f elf lab7-1.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[apchesnokov@fedora lab07]$ ./lab7-1
j
[apchesnokov@fedora lab07]$ nasm -f elf lab7-1.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[apchesnokov@fedora lab07]$ ./lab7-1

[apchesnokov@fedora lab07]$
```

Рис. 1.4: Работа программы

4. Для работы с числами в файле `in_out.asm` существуют подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы из Листинга 7.1 с использованием этих функций. (рис. 1.5, 1.6)

A code editor window with a light blue header. The title bar shows 'Открыть' (Open) with a dropdown arrow, a '+' icon, the filename 'lab7-2.asm', and the path '~/work/study/2022-2023/...ютепа/arch-pc/labs/lab07'. Below the title bar are two tabs: 'report.md' and 'lab7-2.asm'. The 'lab7-2.asm' tab is active, showing the following assembly code:

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

Рис. 1.5: Пример программы


```
[apchesnokov@fedora lab07]$ nasm -f elf lab7-2.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[apchesnokov@fedora lab07]$ ./lab7-2
106
[apchesnokov@fedora lab07]$
```

Рис. 1.6: 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 программы из листинга 7.1, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число.

5. Аналогично предыдущему примеру изменим символы на числа. (рис. 1.7, 1.8)

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

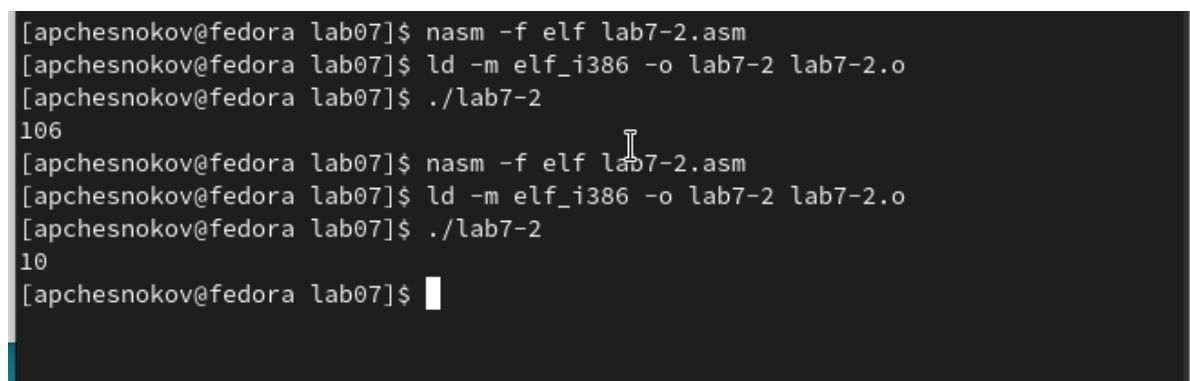


```
lab7-2.asm
~/work/study/2022-2023/...ютеpa/arch-pc/labs/lab07

report.md lab7-2.asm x

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

Рис. 1.7: Пример программы



```
[apchesnokov@fedora lab07]$ nasm -f elf lab7-2.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[apchesnokov@fedora lab07]$ ./lab7-2
106
[apchesnokov@fedora lab07]$ nasm -f elf lab7-2.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[apchesnokov@fedora lab07]$ ./lab7-2
10
[apchesnokov@fedora lab07]$
```

Рис. 1.8: Работа программы

Заменяем функцию `iprintLF` на `iprint` и создадим исполняемый файл и запустим. Чем отличается вывод функций `iprintLF` и `iprint`? - Вывод отличается что нет переноса строки. (рис. 1.9)

```

[apchesnokov@fedora lab07]$ nasm -f elf lab7-2.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[apchesnokov@fedora lab07]$ ./lab7-2
106
[apchesnokov@fedora lab07]$ nasm -f elf lab7-2.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[apchesnokov@fedora lab07]$ ./lab7-2
10
[apchesnokov@fedora lab07]$ nasm -f elf lab7-2.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[apchesnokov@fedora lab07]$ ./lab7-2
10[apchesnokov@fedora lab07]$ █

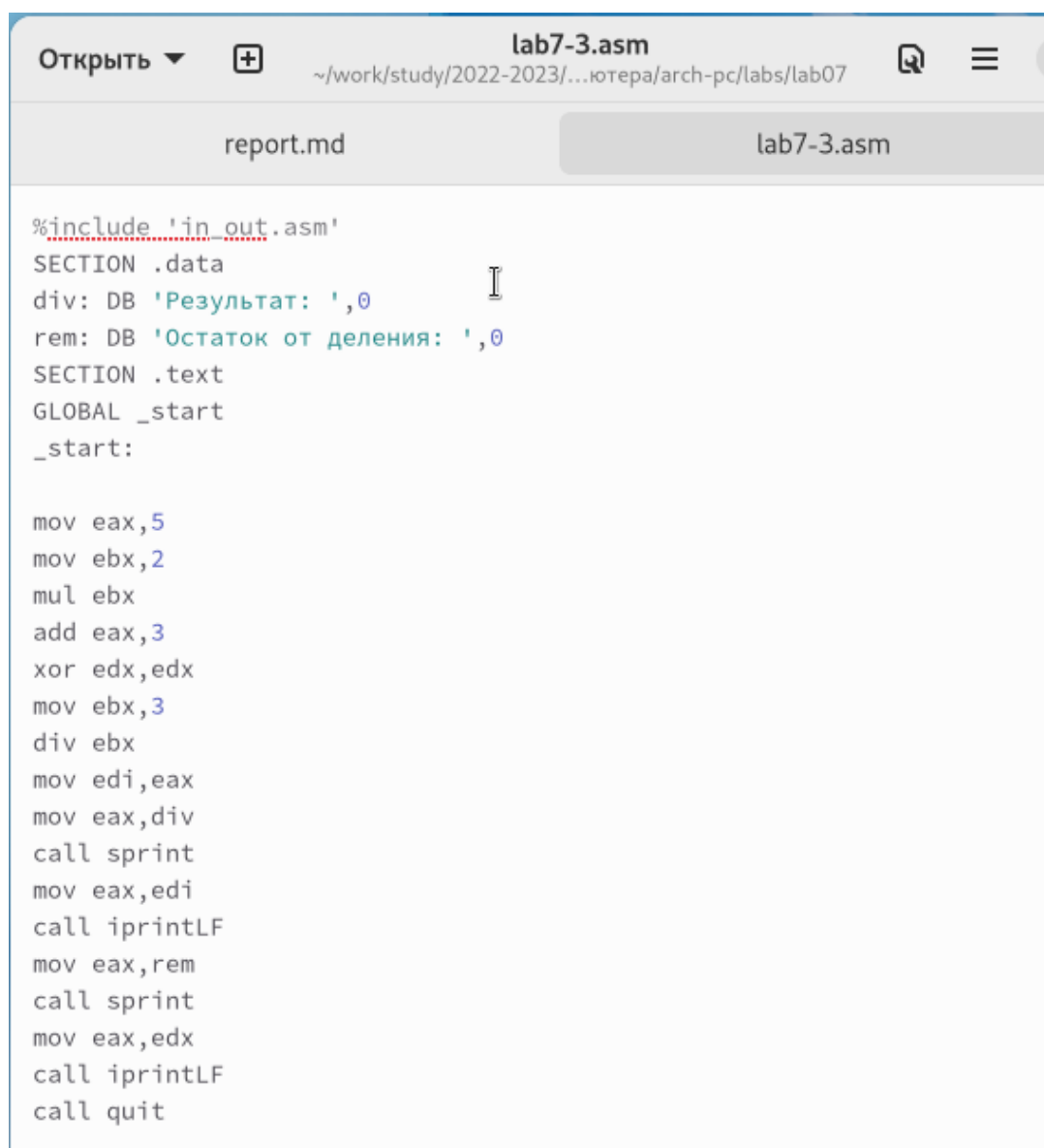
```

Рис. 1.9: Работа программы

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

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

. (рис. 1.10, рис. 1.11)



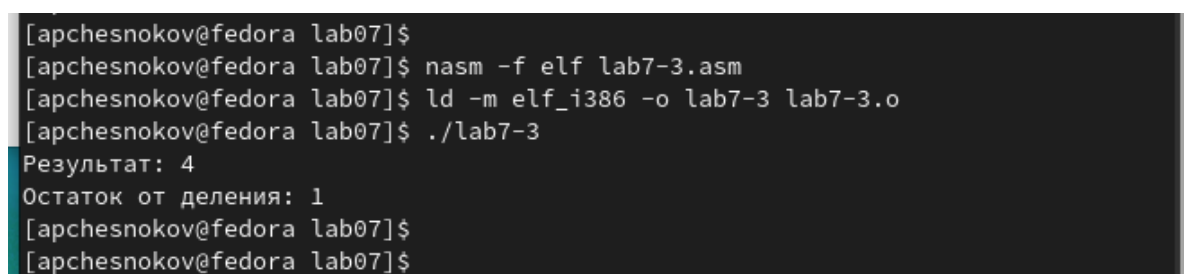
```
Открыть ▾ + lab7-3.asm
~/work/study/2022-2023/...ютепа/arch-pc/labs/lab07

report.md 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,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. 1.10: Пример программы



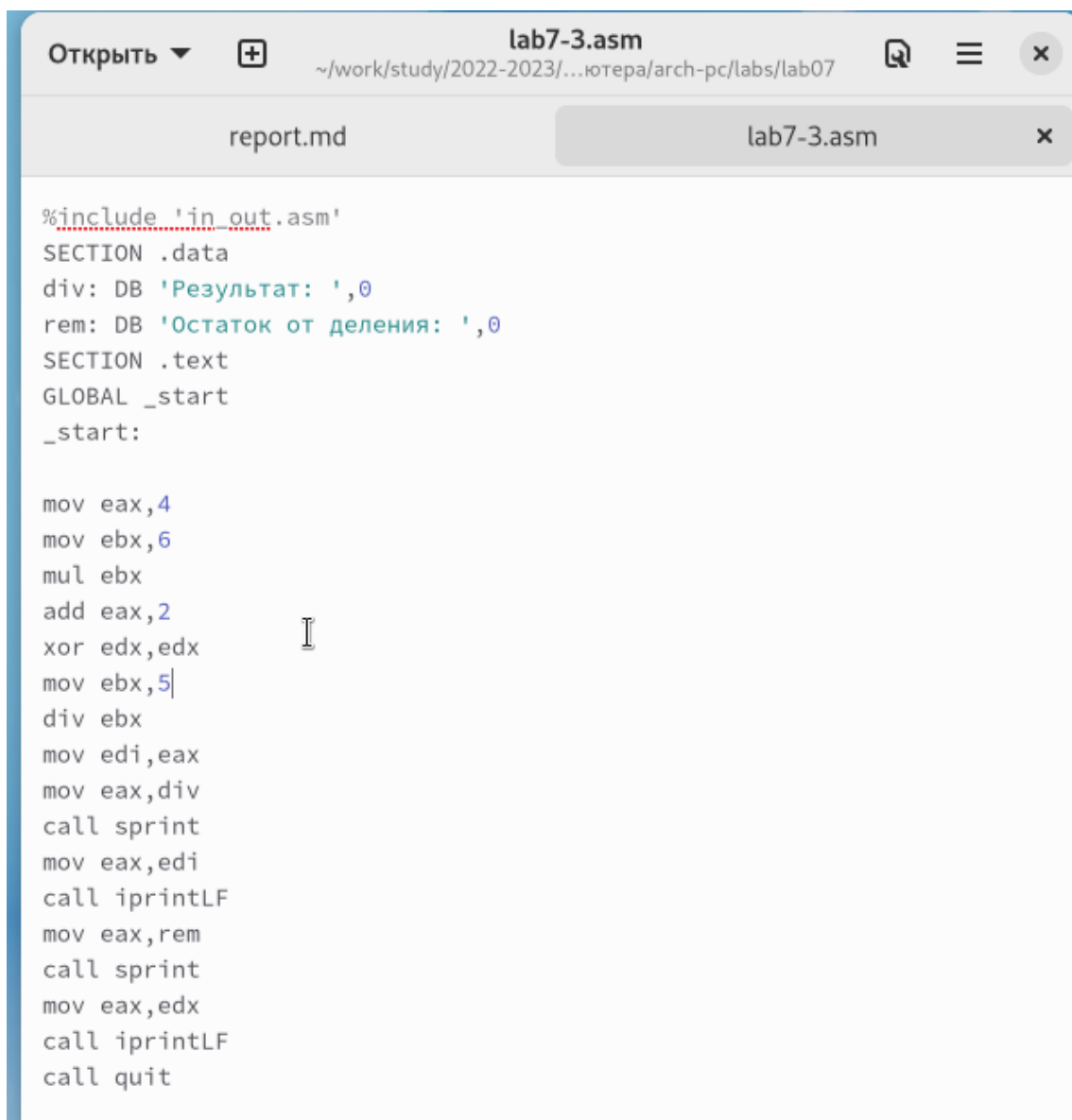
```
[archesnokov@fedora lab07]$
[archesnokov@fedora lab07]$ nasm -f elf lab7-3.asm
[archesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[archesnokov@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
[archesnokov@fedora lab07]$
[archesnokov@fedora lab07]$
```

Рис. 1.11: Работа программы

Изменим текст программы для вычисления выражения

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

. Создадим исполняемый файл и проверьте его работу. (рис. 1.12, рис. 1.13)



```
Открыть ▾ + lab7-3.asm ~/work/study/2022-2023/...ютепа/arch-pc/labs/lab07
report.md lab7-3.asm x

%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. 1.12: Пример программы

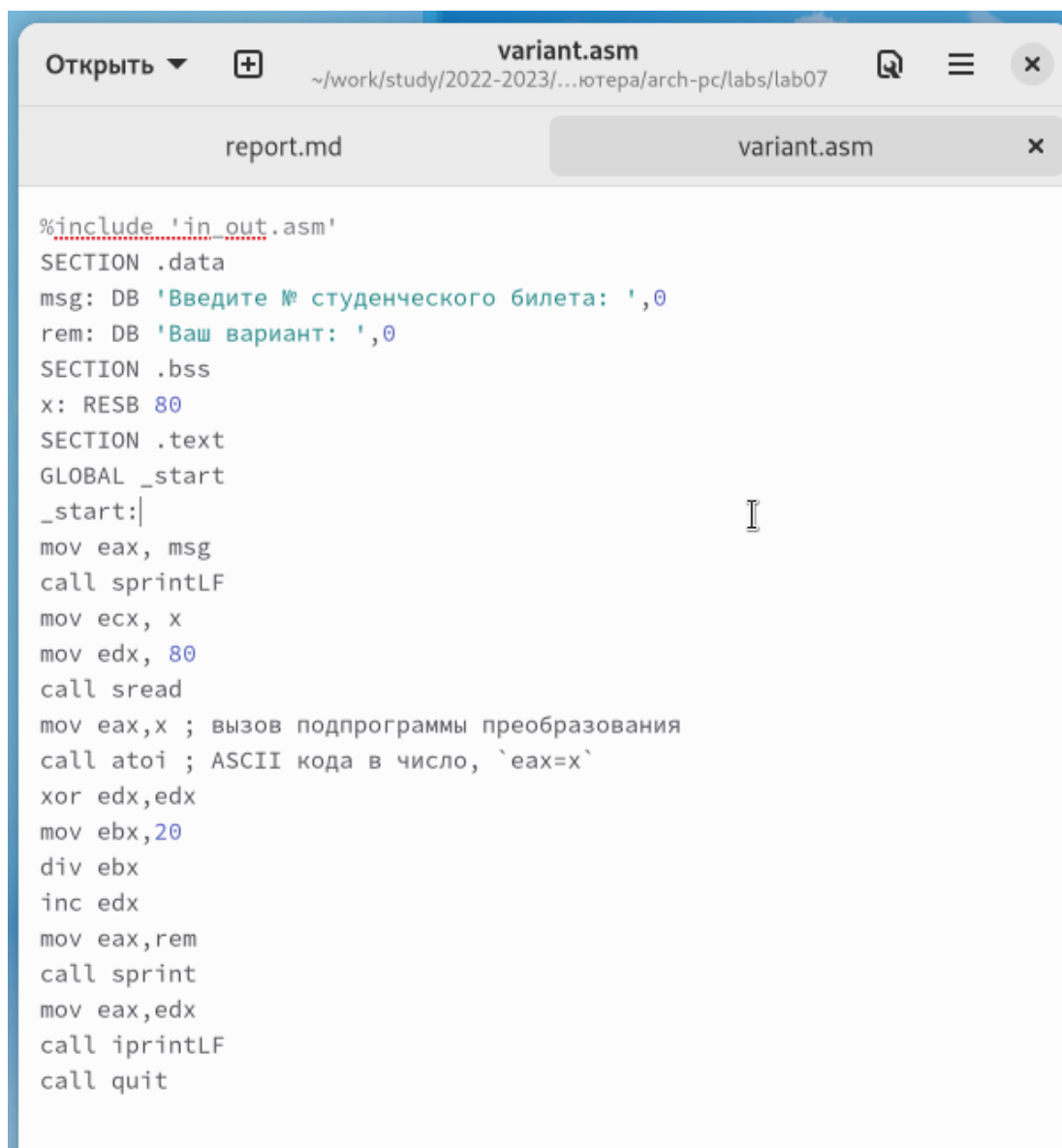
```

[apchesnokov@fedora lab07]$
[apchesnokov@fedora lab07]$ nasm -f elf lab7-3.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[apchesnokov@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
[apchesnokov@fedora lab07]$
[apchesnokov@fedora lab07]$
[apchesnokov@fedora lab07]$ nasm -f elf lab7-3.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[apchesnokov@fedora lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1
[apchesnokov@fedora lab07]$

```

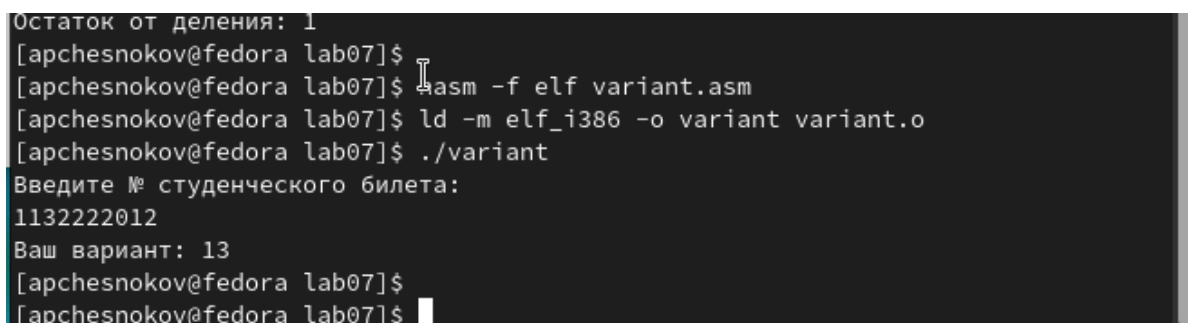
Рис. 1.13: Работа программы

7. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму: (рис. 1.14, рис. 1.15)



```
%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 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, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit
```

Рис. 1.14: Пример программы



```
Остаток от деления: 1
[apchesnokov@fedora lab07]$
[apchesnokov@fedora lab07]$ nasm -f elf variant.asm
[apchesnokov@fedora lab07]$ ld -m elf_i386 -o variant variant.o
[apchesnokov@fedora lab07]$ ./variant
Введите № студенческого билета:
1132222012
Ваш вариант: 13
[apchesnokov@fedora lab07]$
[apchesnokov@fedora lab07]$
```

Рис. 1.15: Работа программы

- Какие строки листинга 7.4 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? `mov ecx, x` `mov edx, 80` `call sread`

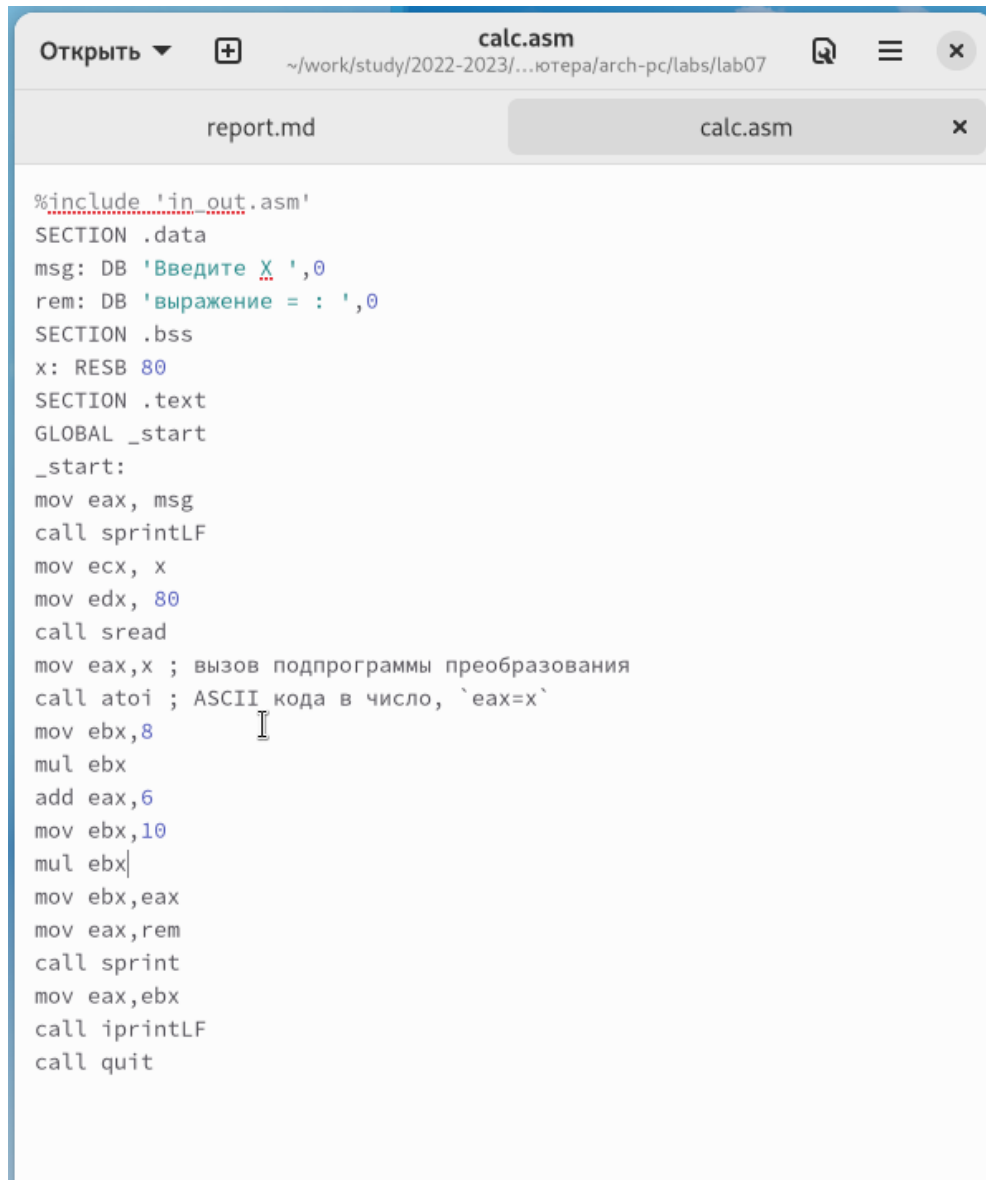
Считывает значение студбилета в переменную X из консоли

- Для чего используется инструкция “`call atoi`”? – эта подпрограмма переводит введенные символы в числовой формат
 - Какие строки листинга 7.4 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 7.4 отвечают за вывод на экран результата вычисления? `mov eax,edx` – результат перекладывается в регистр `eax` `call iprintLF` – вызов подпрограммы вывода
8. Напишем 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 $f(x)$ выберем из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создадим исполняемый файл и проверим его работу для x_1 и x_2 из 6.3. (рис. 1.16, рис. 1.17)

Получили вариант 13 -

$$(8x + 6) * 10$$

для $x=1$ и 4



```
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите X ',0
rem: DB 'выражение = : ',0
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`
mov ebx, 8
mul ebx
add eax, 6
mov ebx, 10
mul ebx
mov ebx, eax
mov eax, rem
call sprintf
mov eax, ebx
call iprintLF
call quit
```

Рис. 1.16: Пример программы

```
[archesnokov@fedora lab07]$  
[archesnokov@fedora lab07]$ nasm -f elf calc.asm  
[archesnokov@fedora lab07]$ ld -m elf_i386 -o calc calc.o  
[archesnokov@fedora lab07]$ ./calc  
Введите X  
1  
выражение = : 140  
[archesnokov@fedora lab07]$ ./calc  
Введите X  
4  
выражение = : 380  
[archesnokov@fedora lab07]$
```

Рис. 1.17: Работа программы

2 Выводы

Изучили работу с арифметическими операциями.