

Отчёт по лабораторной работе 6

дисциплина: Архитектура компьютера

Кайд Омар Мохамад НБИбд-03-23

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	23

Список иллюстраций

2.1	Программа lab6-1.asm	7
2.2	Запуск программы lab6-1.asm	7
2.3	Программа lab6-1.asm	8
2.4	Запуск программы lab6-1.asm	9
2.5	Программа lab6-2.asm	10
2.6	Запуск программы lab6-2.asm	10
2.7	Программа lab6-2.asm	11
2.8	Запуск программы lab6-2.asm	11
2.9	Программа lab6-2.asm	12
2.10	Запуск программы lab6-2.asm	13
2.11	Программа lab6-3.asm	14
2.12	Запуск программы lab6-3.asm	15
2.13	Программа lab6-3.asm	16
2.14	Запуск программы lab6-3.asm	17
2.15	Программа variant.asm	18
2.16	Запуск программы variant.asm	19
2.17	Программа calc.asm	21
2.18	Запуск программы calc.asm	22

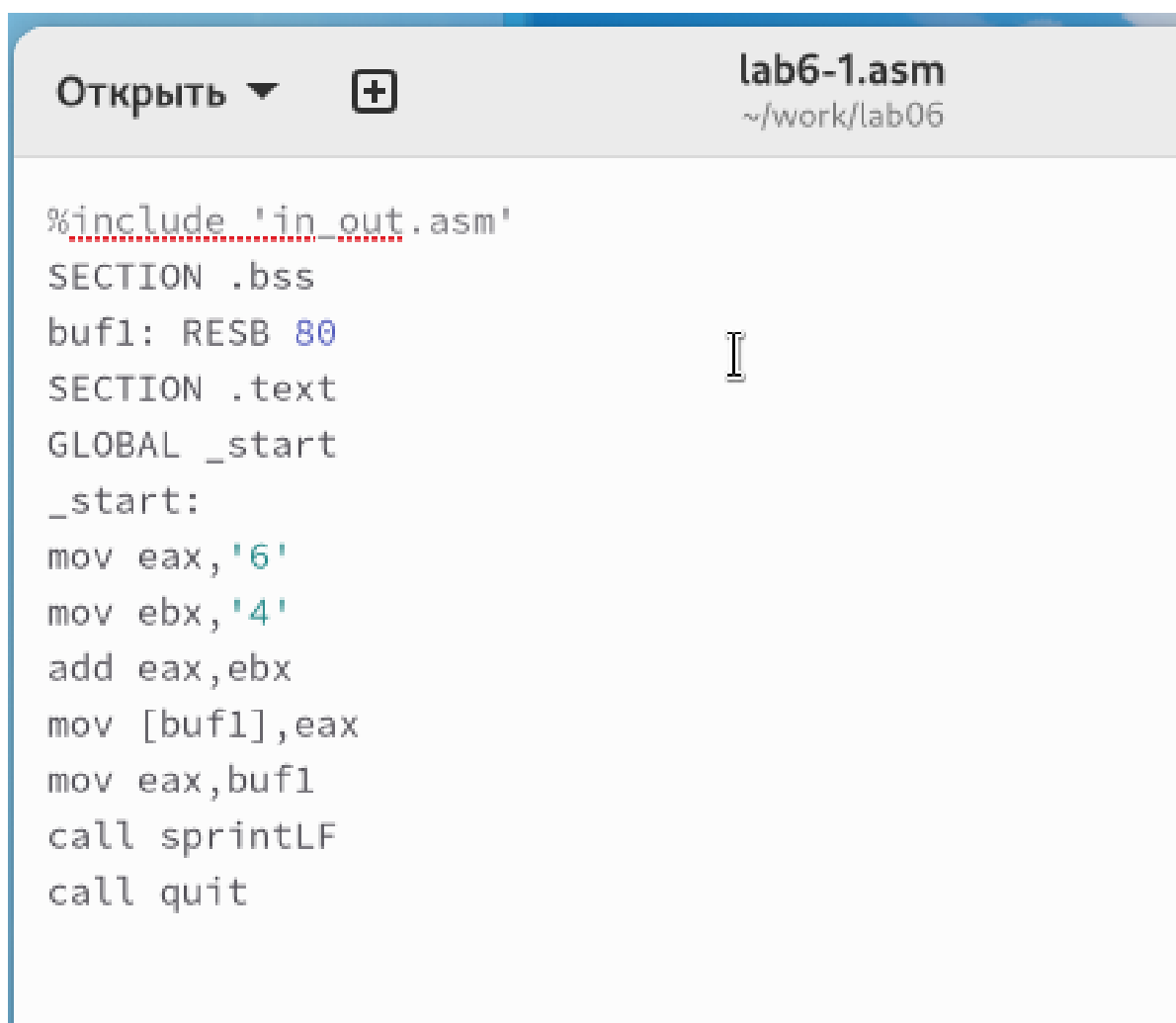
Список таблиц

1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

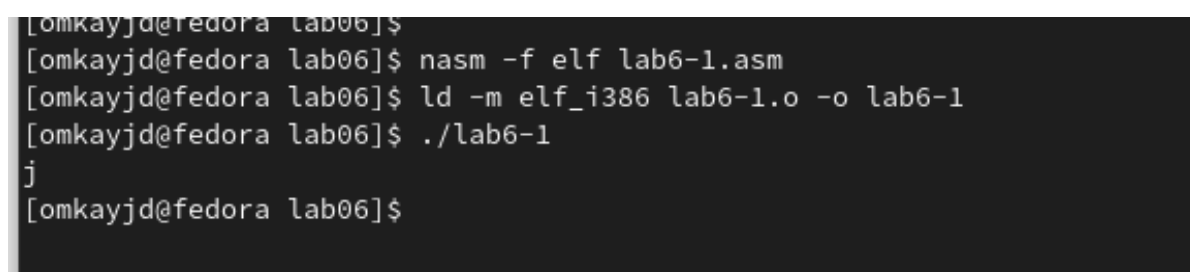
1. Создал каталог для программ лабораторной работы № 6, перешел в него и создал файл lab6-1.asm.
2. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистрах еах.



```
Открыть ▼  +  lab6-1.asm
~/work/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: Программа lab6-1.asm

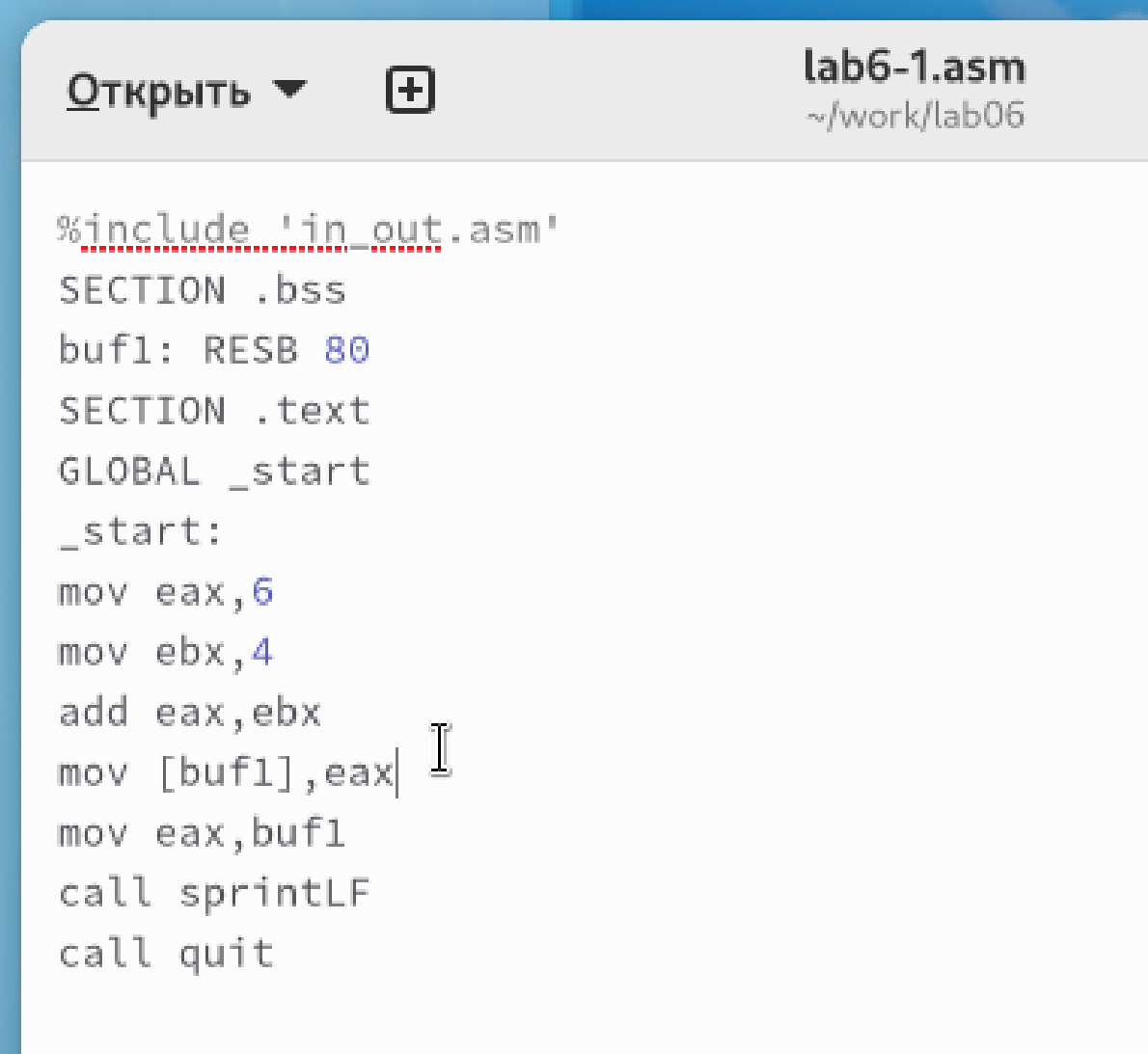


```
[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$ nasm -f elf lab6-1.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-1.o -o lab6-1
[omkayjd@fedora lab06]$ ./lab6-1
j
[omkayjd@fedora lab06]$
```

Рис. 2.2: Запуск программы lab6-1.asm

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

числа.



```
%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
```

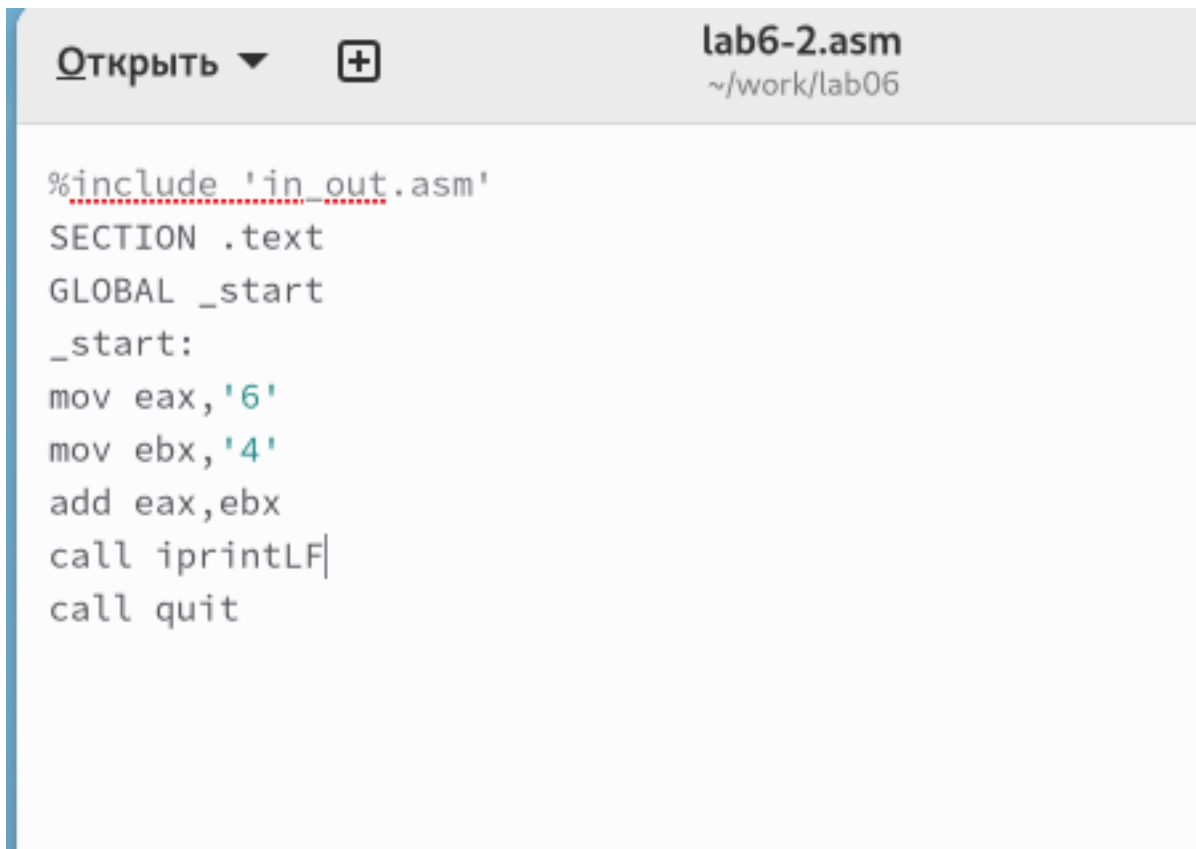
Рис. 2.3: Программа lab6-1.asm


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

Рис. 2.4: Запуск программы lab6-1.asm

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

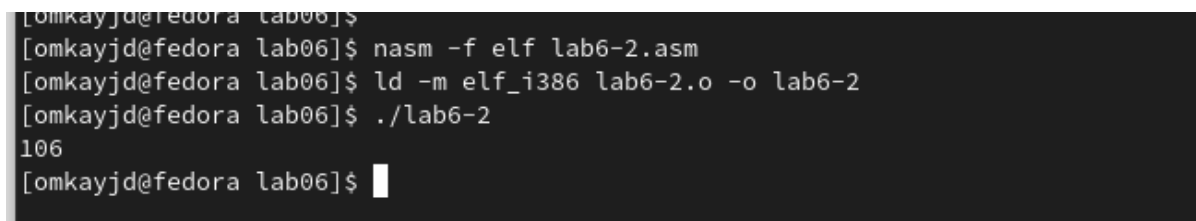
4. Как отмечалось выше, для работы с числами в файле in_out.asm реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразовал текст программы с использованием этих функций.



```
Открыть ▾ + lab6-2.asm
~/work/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: Программа lab6-2.asm

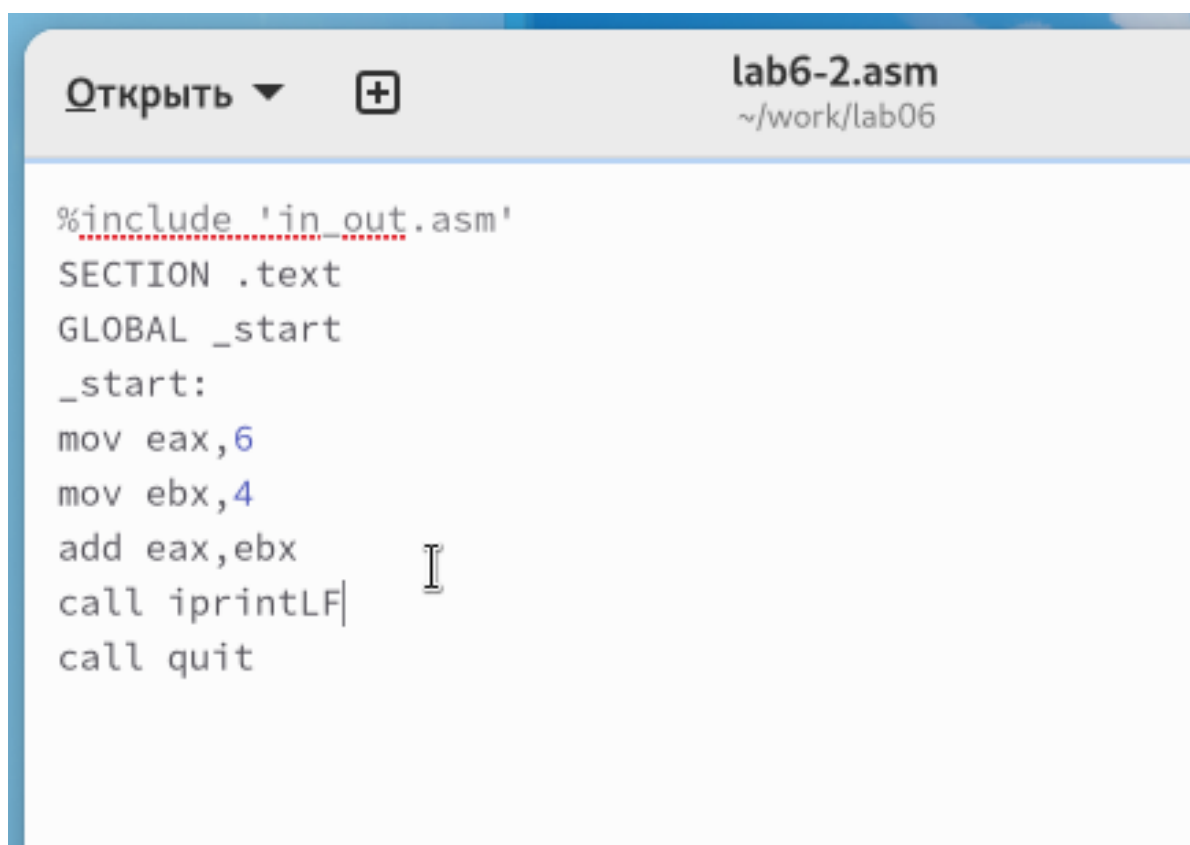


```
[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$ nasm -f elf lab6-2.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[omkayjd@fedora lab06]$ ./lab6-2
106
[omkayjd@fedora lab06]$
```

Рис. 2.6: Запуск программы lab6-2.asm

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.

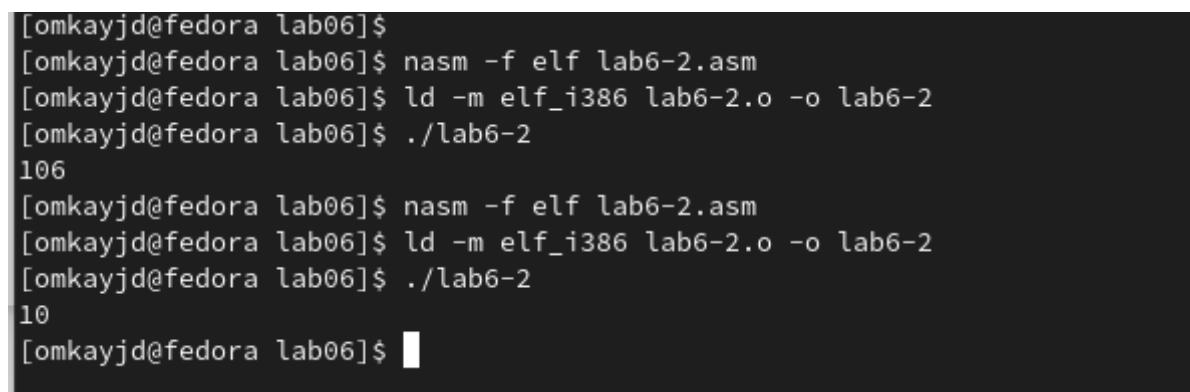


```
Открыть ▾ + lab6-2.asm
~/work/lab06

%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: Программа lab6-2.asm

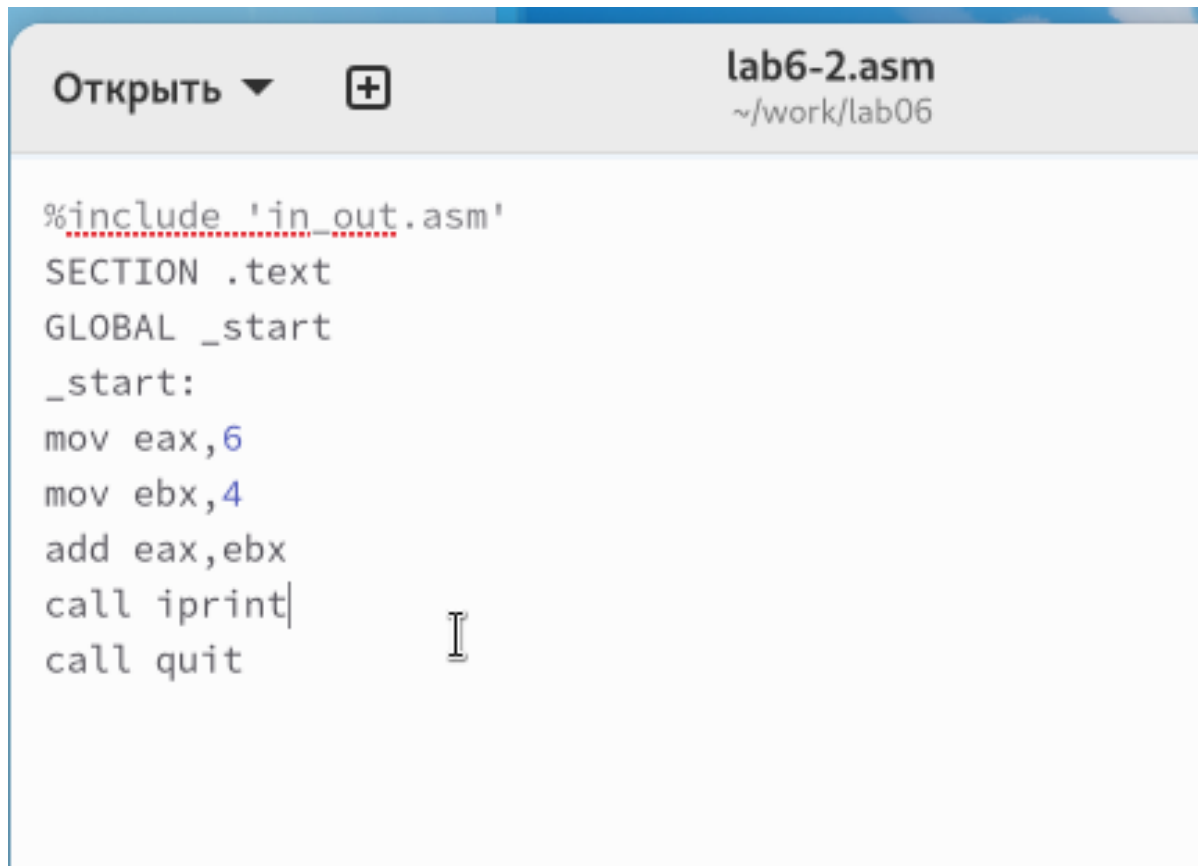
Функция `iprintLF` позволяет вывести число и операндами были числа (а не коды символов). Поэтому получаем число 10.



```
[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$ nasm -f elf lab6-2.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[omkayjd@fedora lab06]$ ./lab6-2
106
[omkayjd@fedora lab06]$ nasm -f elf lab6-2.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[omkayjd@fedora lab06]$ ./lab6-2
10
[omkayjd@fedora lab06]$
```

Рис. 2.8: Запуск программы lab6-2.asm

Заменяю функцию `iprintLF` на `iprint`. Вывод отличается тем, что нет переноса строки.



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

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

Рис. 2.9: Программа lab6-2.asm


```

[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$ nasm -f elf lab6-2.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[omkayjd@fedora lab06]$ ./lab6-2
106
[omkayjd@fedora lab06]$ nasm -f elf lab6-2.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[omkayjd@fedora lab06]$ ./lab6-2
10
[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$ nasm -f elf lab6-2.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-2.o -o lab6-2
[omkayjd@fedora lab06]$ ./lab6-2
10[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$

```

Рис. 2.10: Запуск программы lab6-2.asm

6. В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения $f(x) = (5 * 2 + 3)/3$.

Открыть ▾  lab6-3.asm
~/work/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. 2.11: Программа lab6-3.asm

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

Рис. 2.12: Запуск программы lab6-3.asm

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



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

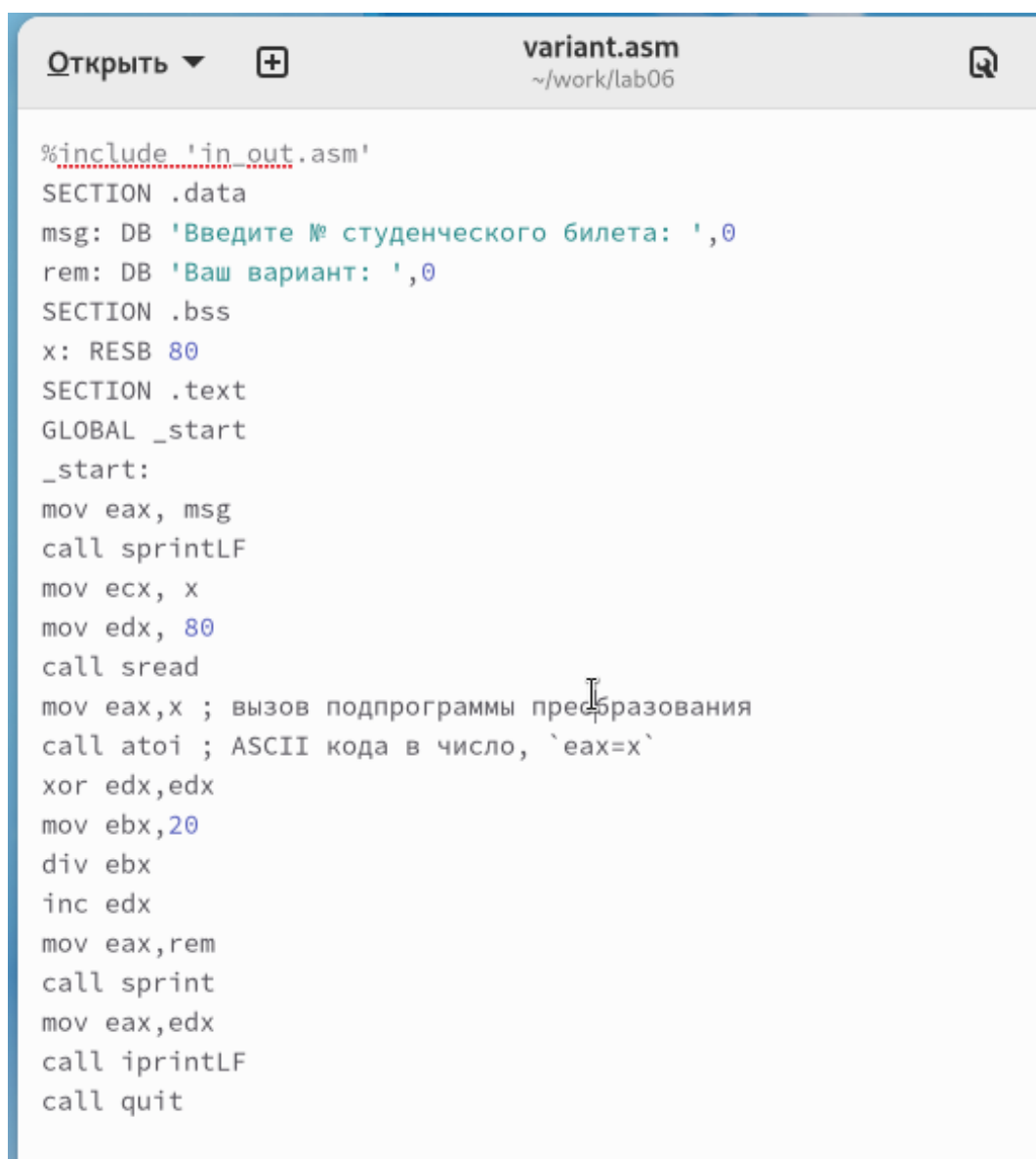
mov eax,4
mov ebx,6
mul ebx
add eax,2
xor edx,edx
mov ebx,5
div ebx
mov edi,eax
mov eax,div
call sprint
mov eax,edi
call iprintLF
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit
```

Рис. 2.13: Программа lab6-3.asm


```
[omkayjd@fedora lab06]$ nasm -f elf lab6-3.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-3.o -o lab6-3
[omkayjd@fedora lab06]$ ./lab6-3
Результат: 4
Остаток от деления: 1
[omkayjd@fedora lab06]$
[omkayjd@fedora lab06]$ nasm -f elf lab6-3.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 lab6-3.o -o lab6-3
[omkayjd@fedora lab06]$ ./lab6-3
Результат: 5
Остаток от деления: 1
[omkayjd@fedora lab06]$
```

Рис. 2.14: Запуск программы lab6-3.asm

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



```
%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. 2.15: Программа variant.asm

```
[omkayjd@fedora lab06]$  
[omkayjd@fedora lab06]$ nasm -f elf variant.asm  
[omkayjd@fedora lab06]$ ld -m elf_i386 variant.o -o variant  
[omkayjd@fedora lab06]$ ./variant  
Введите № студенческого билета:  
1032230167  
Ваш вариант: 8  
[omkayjd@fedora lab06]$
```

Рис. 2.16: Запуск программы variant.asm

ответы на вопросы

1. Какие строки листинга отвечают за вывод на экран сообщения ‘Ваш вариант:’?

Значение переменной с фразой ‘Ваш вариант:’ перекладывается в регистры `eax` с помощью строки `mov eax, rem`. Вызывается подпрограмма `sprint` для вывода строки.

2. Для чего используются следующие инструкции?

```
mov ecx, x  
mov edx, 80  
call sread
```

Инструкция `mov ecx, x` перемещает значение переменной `X` в регистр `ecx`.

Инструкция `mov edx, 80` устанавливает значение 80 в регистр `edx`.

Инструкция `call sread` вызывает подпрограмму для чтения значения с консоли.

3. Для чего используется инструкция “call atoi”?

Инструкция `call atoi` вызывает подпрограмму, которая преобразует введенные символы в числовой формат.

4. Какие строки листинга отвечают за вычисления варианта?

Инструкция `xor edx, edx` обнуляет регистр `edx`.

Инструкция `mov ebx, 20` устанавливает значение 20 в регистр `ebx`.

Инструкция `div ebx` выполняет деление номера студенческого билета на 20.

Инструкция `inc edx` увеличивает значение регистра `edx` на 1.

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

Остаток от деления записывается в регистр `edx`.

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

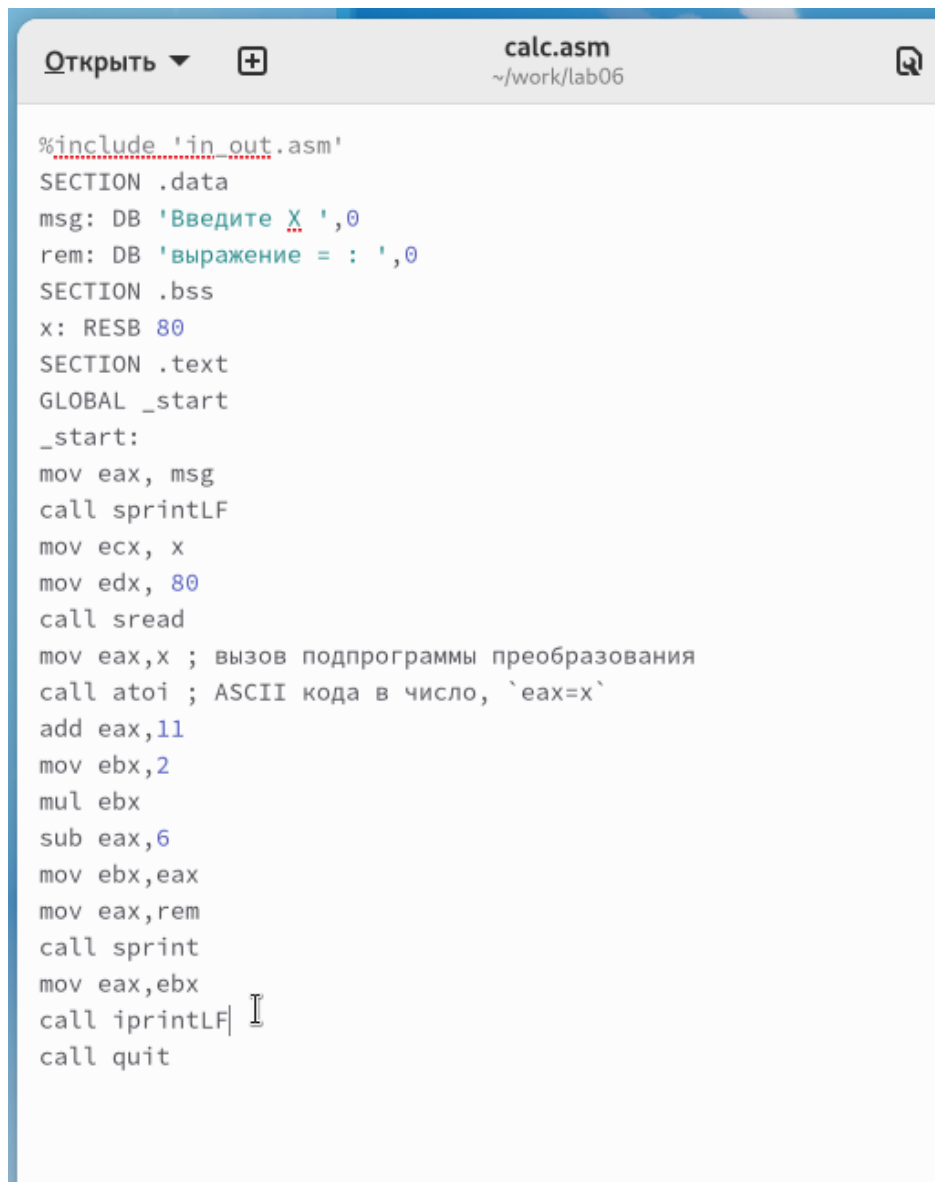
Инструкция `inc edx` используется для увеличения значения регистра `edx` на 1. В данном случае, она используется для выполнения формулы вычисления варианта, где требуется добавить 1 к остатку от деления.

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

Результат вычислений перекладывается в регистр `eax` с помощью строки `mov eax, edx`. Вызывается подпрограмма `iprintLF` для вывода результата на экран.

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 8 - $-(11+x)^2-6$ для $x = 1, x = 9$



```
%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 sprintLF
mov ecx, x
mov edx, 80
call sread
mov eax,x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, `eax=x`
add eax,11
mov ebx,2
mul ebx
sub eax,6
mov ebx,eax
mov eax,rem
call sprint
mov eax,ebx
call iprintLF
call quit
```

Рис. 2.17: Программа calc.asm

```
[omkayjd@fedora lab06]$ nasm -f elf calc.asm
[omkayjd@fedora lab06]$ ld -m elf_i386 calc.o -o calc
[omkayjd@fedora lab06]$ ./calc
Введите X
1
выражение = : 18
[omkayjd@fedora lab06]$ ./calc
Введите X
9
выражение = : 34
[omkayjd@fedora lab06]$
```

Рис. 2.18: Запуск программы calc.asm

3 Выводы

Изучили работу с арифметическими операциями.