Отчёт по лабораторной работе 6

Архитектура компьютеров

Дмитраков Михаил Алексеевич

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

2.1 Символьные и численные данные в NASM

В начале лабораторной работы я создаю отдельный каталог для программ лабораторной № 6, перехожу в него и формирую файл `lab6-1.asm`.

Далее рассматриваю примеры программ, предназначенных для вывода символов и чисел. Во всех случаях вывод осуществляется на основе значения, находящегося в регистре eax.

В первой программе в регистр eax помещается символ `'6'` с помощью инструкции `mov eax, '6'`, а в регистр ebx — символ `'4'` (`mov ebx, '4'`). Затем выполняется сложение содержимого регистров (`add eax, ebx`), и результат сохраняется в eax, после чего выводится на экран.

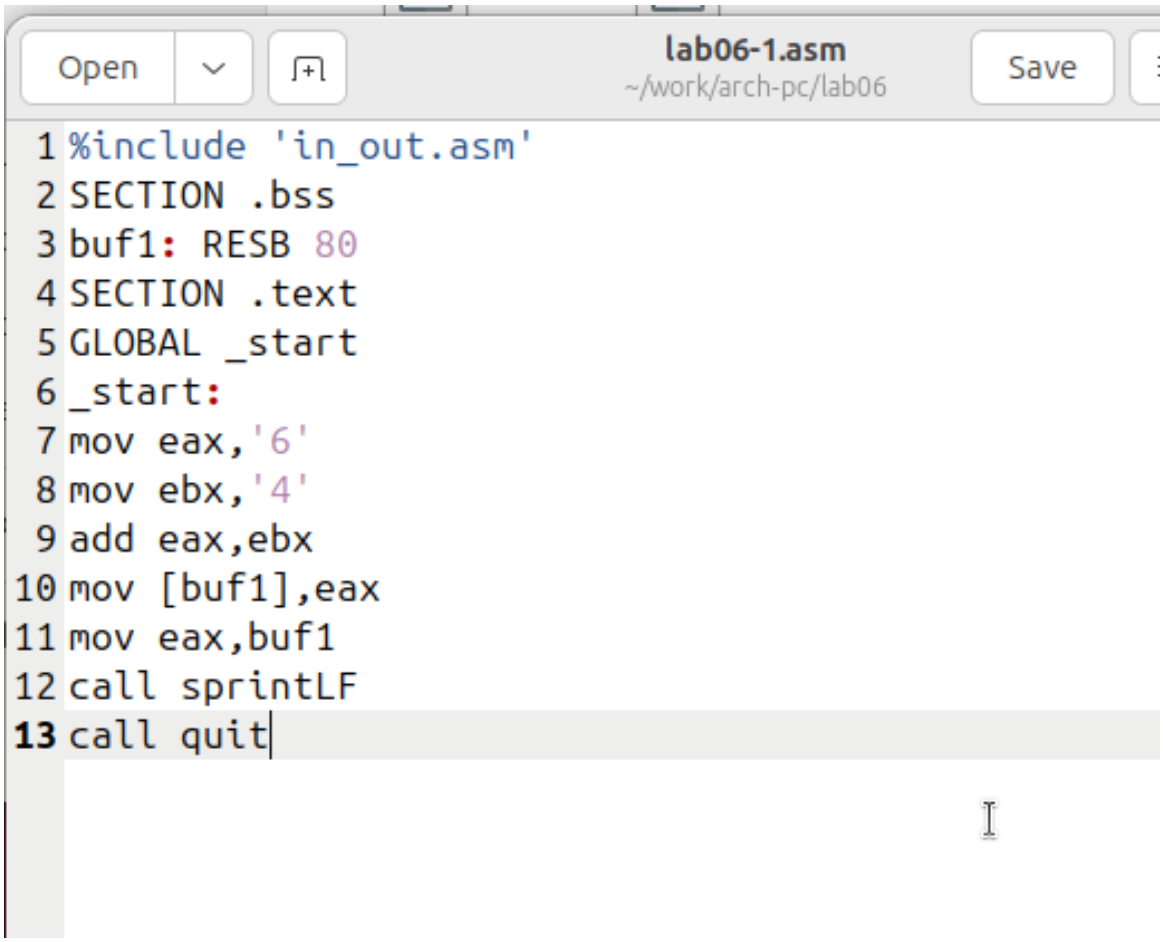

Рисунок 2.1: Код программы lab6-1.asm

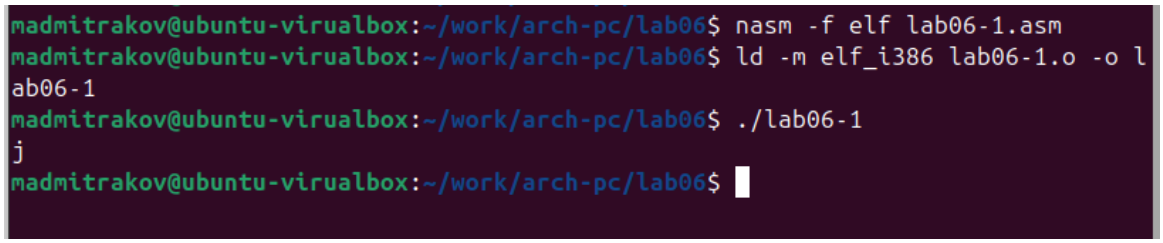

Рисунок 2.2: Запуск программы lab6-1.asm

Ожидаемым результатом является число 10, однако на экране появляется символ `j`. Это объясняется тем, что складываются не числа, а их ASCII-коды: код символа `'6'` равен 54, а символа `'4'` — 52. В результате сложения получается 106, что соответствует символу `j` в таблице ASCII.

Затем программа была изменена: вместо символов в регистры записываются числовые значения.

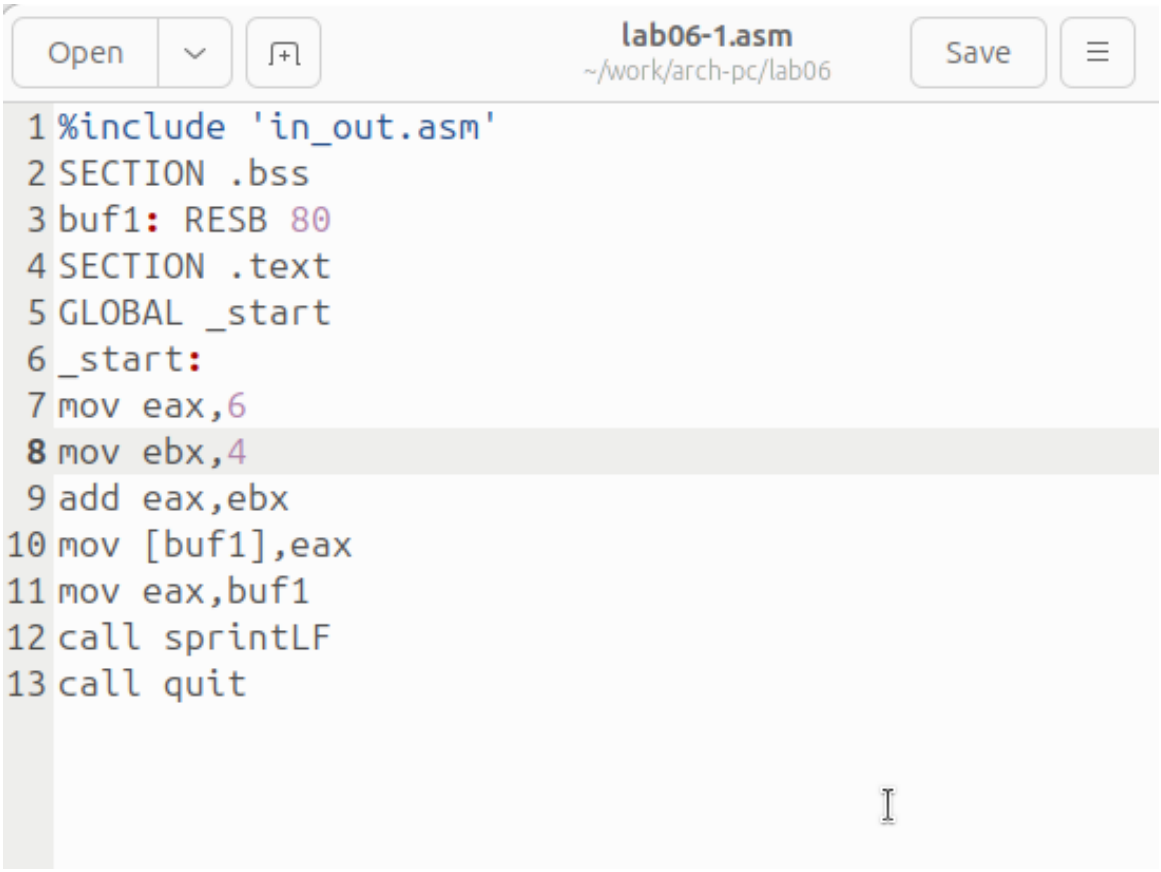

Рисунок 2.3: Код программы lab6-1.asm с числами

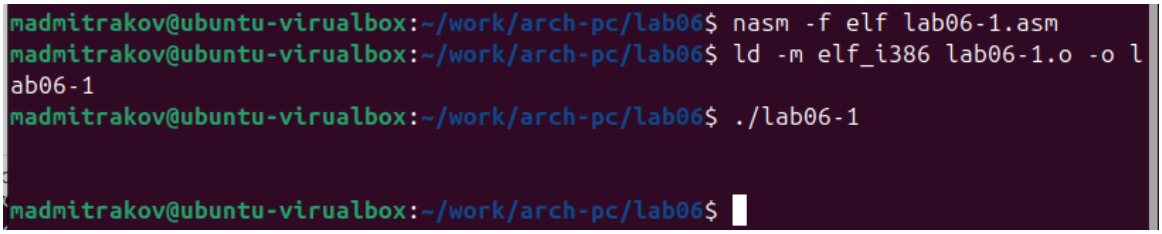

Рисунок 2.4: Запуск программы lab6-1.asm с числами

Однако и в этом случае результатом не является число 10. Выводится символ с кодом 10, который соответствует переводу строки. В терминале он визуально не отображается, но приводит к появлению пустой строки.

Для корректной работы с числовыми значениями используются подпрограммы из файла `in_out.asm`, предназначенные для преобразования ASCII-символов в числа и обратно. Программа была доработана с использованием этих функций.

Рисунок 2.5: Код программы lab6-2.asm

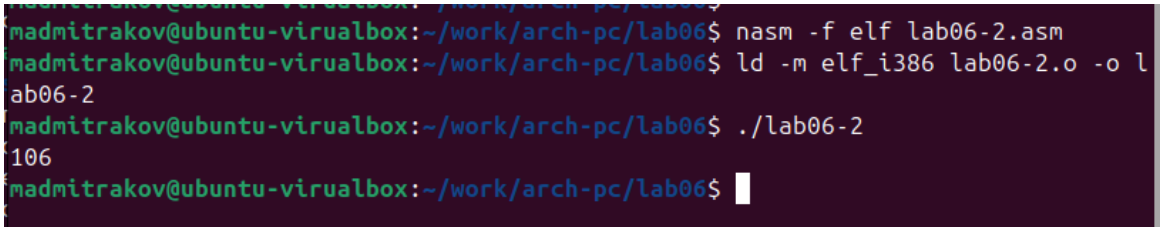

Рисунок 2.6: Запуск программы lab6-2.asm

В результате выполнения данной версии программы выводится число 106. Команда add по-прежнему складывает коды символов `'6'` и `'4'`, однако функция `iprintLF` выводит числовое значение, а не соответствующий символ.

После этого символы снова были заменены на числа.

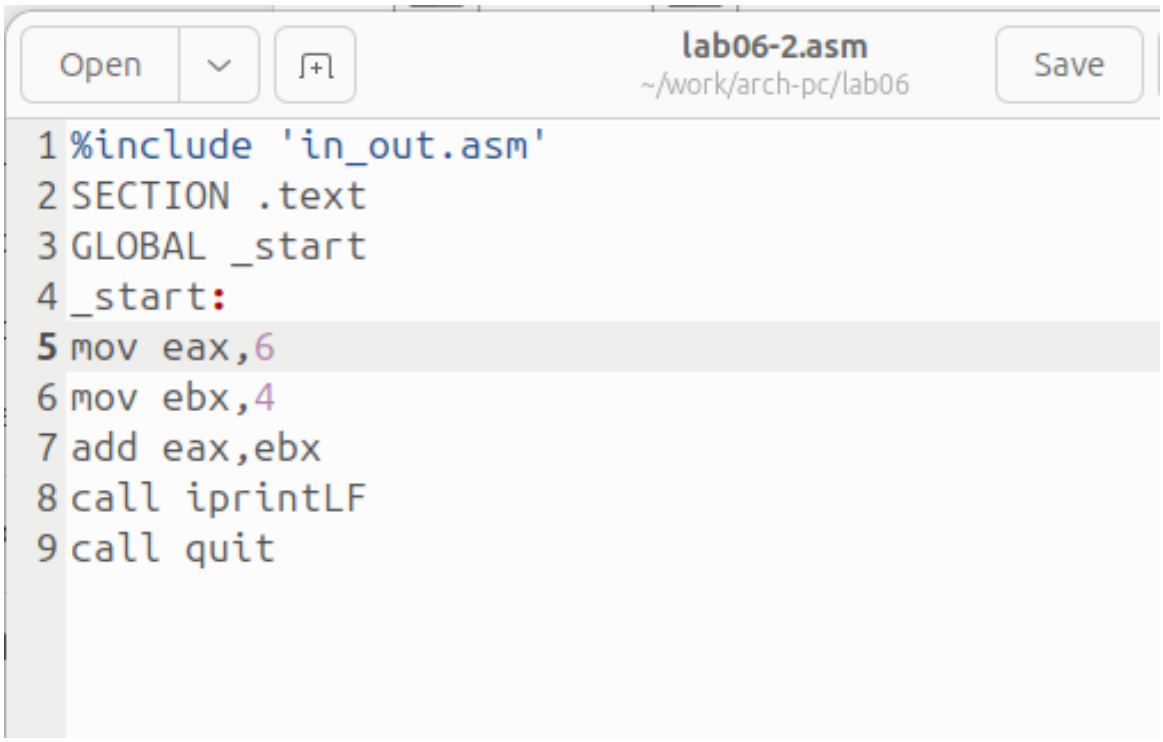

Рисунок 2.7: Код программы lab6-2.asm с числами

В данном случае функция `iprintLF` корректно выводит результат вычислений, так как операндами являются числовые значения. Итоговый результат — число 10.

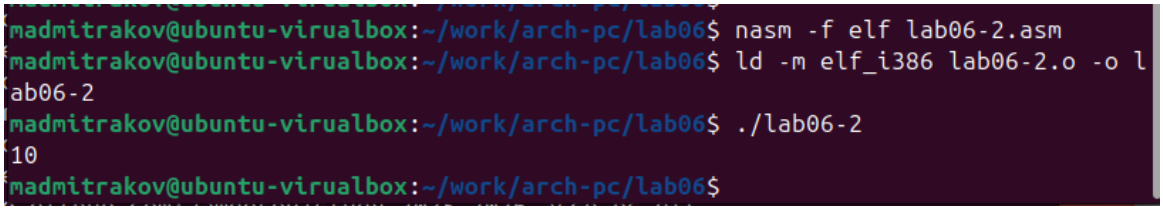

Рисунок 2.8: Запуск программы lab6-2.asm с числами

Затем функция `iprintLF` была заменена на `iprint`. После пересборки и запуска программы вывод отличается отсутствием переноса строки.

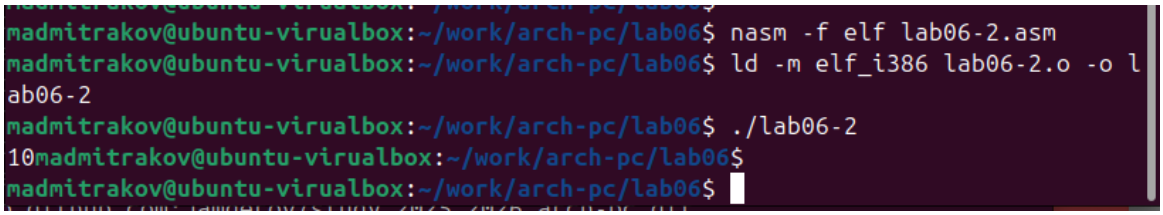

Рисунок 2.9: Запуск программы lab6-2.asm без переноса строки

2.2 Выполнение арифметических операций в NASM

Для демонстрации арифметических операций в NASM рассматривается программа, вычисляющая выражение

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

Open lab06-3.asm ~/work/arch-pc/lab06 Save

```
%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
```

Рисунок 2.10: Код программы lab6-3.asm

```
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$ nasm -f elf lab06-3.asm
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$ ld -m elf_i386 lab06-3.o -o l
ab06-3
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$ ./lab06-3
Результат: 4
Остаток от деления: 1
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$
```

Рисунок 2.11: Запуск программы lab6-3.asm

Далее программа была изменена для вычисления выражения $f(x) = (4 * 6 + 2)/5$. После создания исполняемого файла была проверена корректность его работы.

Open | lab06-3.asm ~/work/arch-pc/lab06 | Save

```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8
9 mov eax,4
10 mov ebx,6
11 mul ebx
12 add eax,2
13 xor edx,edx
14 mov ebx,5
15 div ebx
16 mov edi,eax
17 mov eax,div
18 call sprint
19 mov eax,edi
20 call iprintLF
21 mov eax,rem
22 call sprint
23 mov eax,edx
24 call iprintLF
25 call quit
```

Рисунок 2.12: Код программы lab6-3.asm с новым выражением

```
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$ nasm -f elf lab06-3.asm
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$ ld -m elf_i386 lab06-3.o -o l
ab06-3
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$ ./lab06-3
Результат: 5
Остаток от деления: 1
madmitrakov@ubuntu-virualbox:~/work/arch-pc/lab06$
```

Рисунок 2.13: Запуск программы lab6-3.asm с новым выражением

После этого была рассмотрена программа, предназначенная для вычисления варианта задания по номеру студенческого билета.

Open | variant.asm ~/work/arch-pc/lab06 | Save

```
%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
call atoi
xor edx,edx
mov ebx,20
div ebx
inc edx
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit
```

Рисунок 2.14: Код программы variant.asm

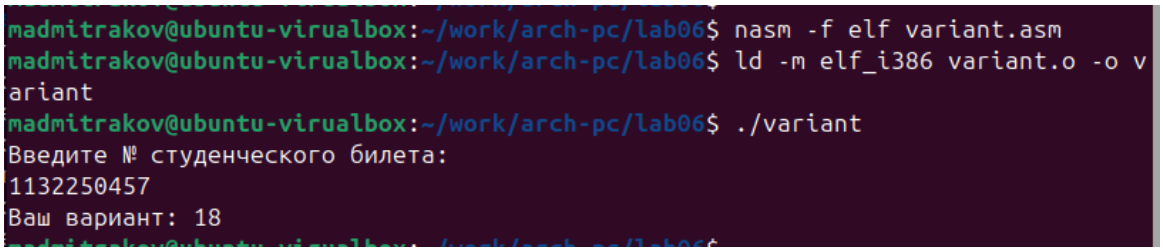

Рисунок 2.15: Запуск программы variant.asm

В данной программе исходное число вводится с клавиатуры в символьном виде. Для дальнейших вычислений введённые данные преобразуются в числовой формат с использованием подпрограммы `atoi` из файла `in_out.asm`.

2.2.1 Ответы на вопросы

1. **Какие строки отвечают за вывод сообщения «Ваш вариант:»?**
   - Инструкция `mov eax, rem` загружает адрес строки с текстом «Ваш вариант:» в регистр `eax`.
   - Инструкция `call sprint` вызывает подпрограмму вывода строки.
2. **Назначение инструкций:**
   - `mov ecx, x` — помещает адрес или значение переменной `x` в регистр `ecx`.
   - `mov edx, 80` — задаёт максимальный размер вводимых данных.
   - `call sread` — вызывает подпрограмму считывания данных с клавиатуры.
3. **Для чего используется `call atoi`?**
   - Данная инструкция преобразует введённую строку символов в числовое значение.
4. **Какие инструкции отвечают за вычисление варианта?**
   - `xor edx, edx` — обнуляет регистр `edx`.

- `mov ebx, 20` — загружает делитель.
- `div ebx` — выполняет деление.
- `inc edx` — увеличивает остаток от деления на единицу.

5. **В какой регистр помещается остаток от деления?**
   - Остаток сохраняется в регистре `edx`.
6. **Назначение инструкции `inc edx`:**
   - Увеличивает значение остатка на 1 в соответствии с формулой определения варианта.
7. **Какие инструкции отвечают за вывод результата?**
   - `mov eax, edx` — передаёт результат в регистр `eax`.
   - `call iprintLF` — выводит числовое значение на экран с переводом строки.

2.3 Задание для самостоятельной работы

В рамках самостоятельного задания была разработана программа для вычисления выражения
$y = f(x)$. Программа выводит формулу, запрашивает ввод значения `x`, вычисляет результат и отображает его на экране.

Согласно полученному варианту № 18 используется формула
$3(x + 10) - 20$ при значениях $x = 1$ и $x = 5$.

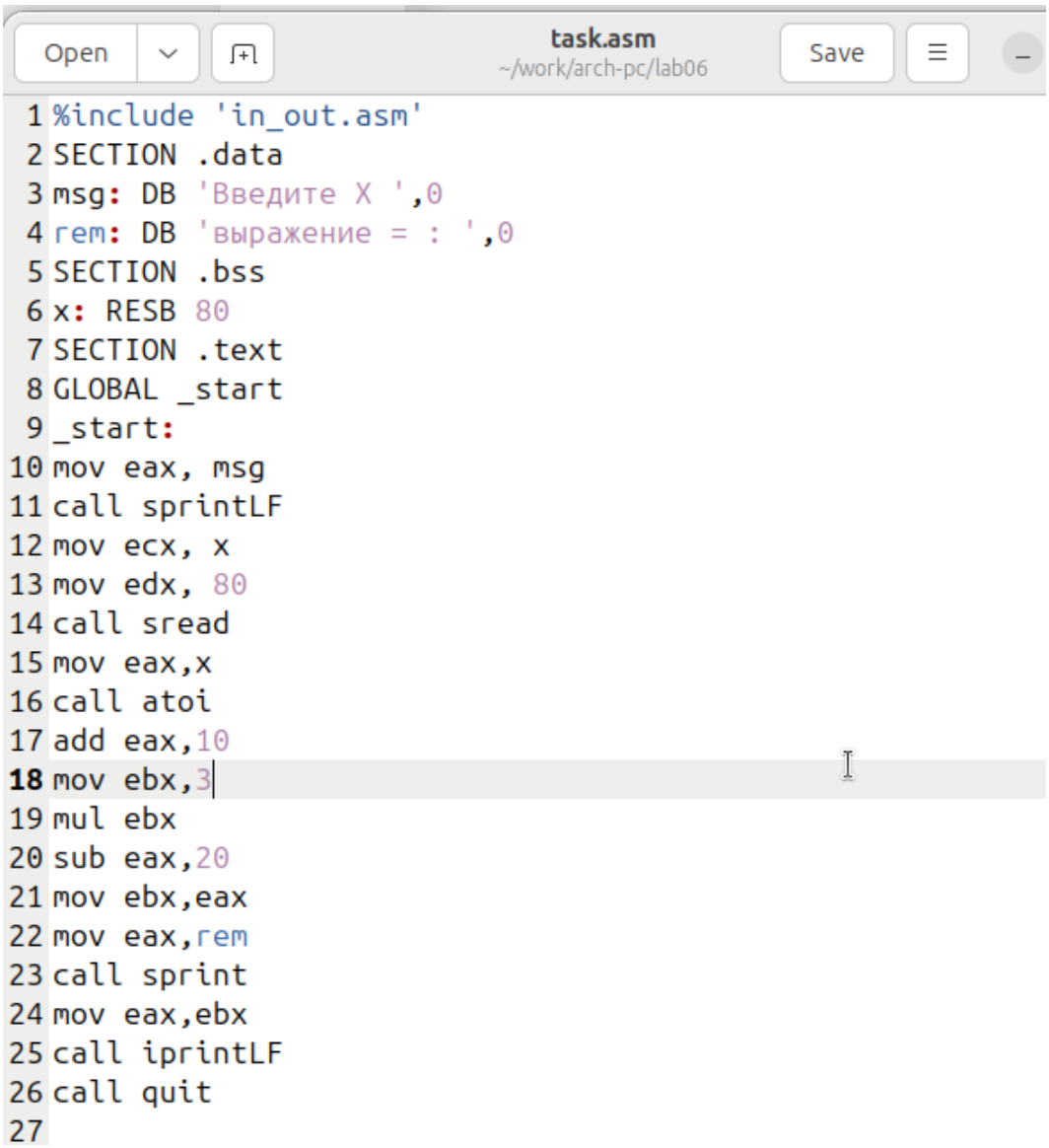

Рисунок 2.16: Код программы task.asm

При значении $x = 1$ результат равен 13.

При значении $x = 5$ результат равен 25.

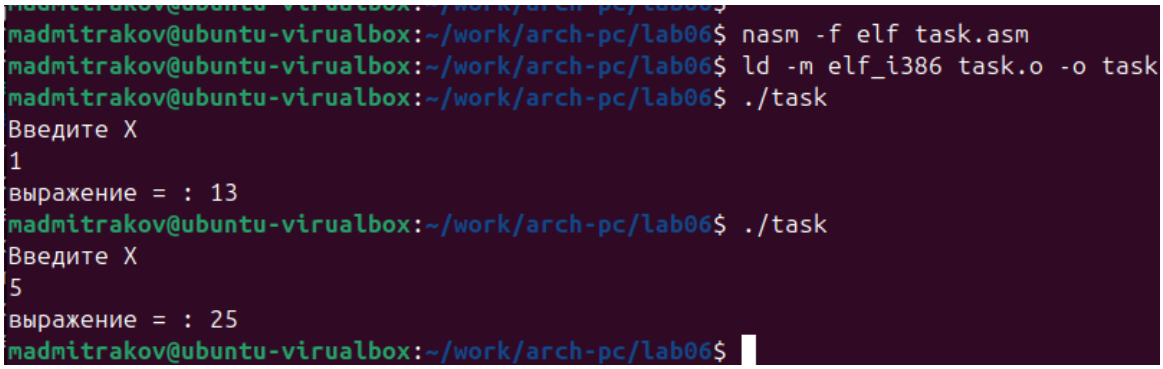

Рисунок 2.17: Запуск программы task.asm

Результаты вычислений соответствуют ожидаемым, что подтверждает корректность работы программы.

3 Выводы

Изучили работу с арифметическими операциями.