РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

Студент: Белакова Полина Вячеславовна

Ст.билет: 1032252589

Группа: НКАбд-01-25

МОСКВА

2025 г

Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

Порядок выполнения лабораторной работы

Символьные и численные данные в NASM

1. Создаю каталог для программ лабораторной работы № 6, перехожу в него и создаю файл lab6-1.asm (рисунок 1).

```
pvbelakova2@localhost-live:~$ mkdir ~/work/arch-pc/lab06
pvbelakova2@localhost-live:~$ cd ~/work/arch-pc/lab06
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ touch lab6-1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab06$
```

Рисунок 1 - создание каталога lab06, перемещение в него и создание файла lab6-1.asm.

2. Ввожу в файл lab6-1.asm текст программы из листинга (рисунок 2).

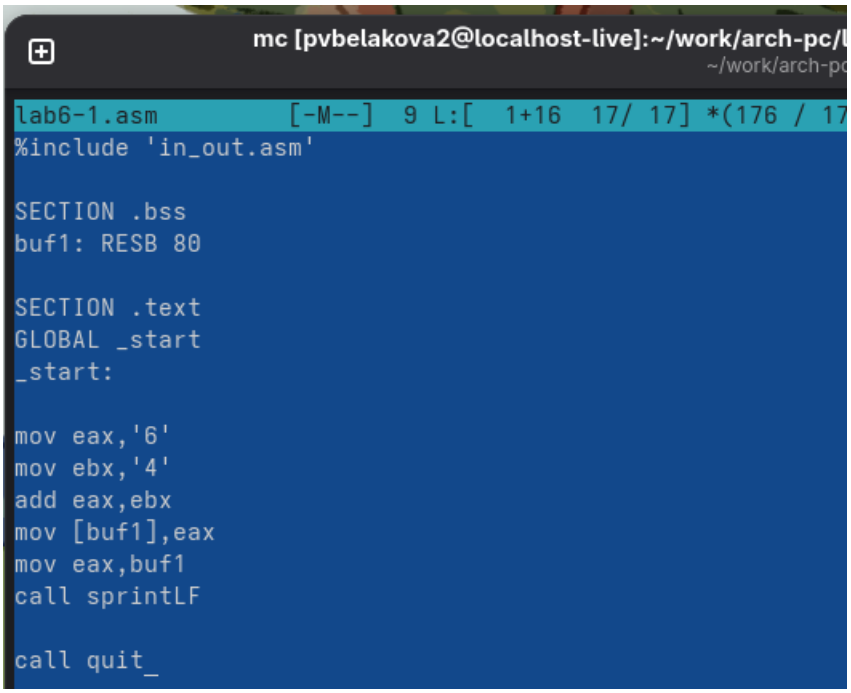

Рисунок 2 - код программы.

Перед созданием исполняемого файла создаю копию файла in_out.asm в каталоге ~/work/arch-pc/lab06, затем ввожу команды для создания объектного и исполняемого файлов и запускаю программу (рисунок 3).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ cp ~/arch-pc/lab05/in_out.asm ~/work/arch-pc/lab06/
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ mc

pvbelakova2@localhost-live:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ./lab6-1
j
pvbelakova2@localhost-live:~/work/arch-pc/lab06$
```

Рисунок 3 - копирование файла, создание объектного и исполняемого файла и запуск программы.

3. Далее изменяю текст программы и вместо символов, записываю в регистры числа (рисунок 4).

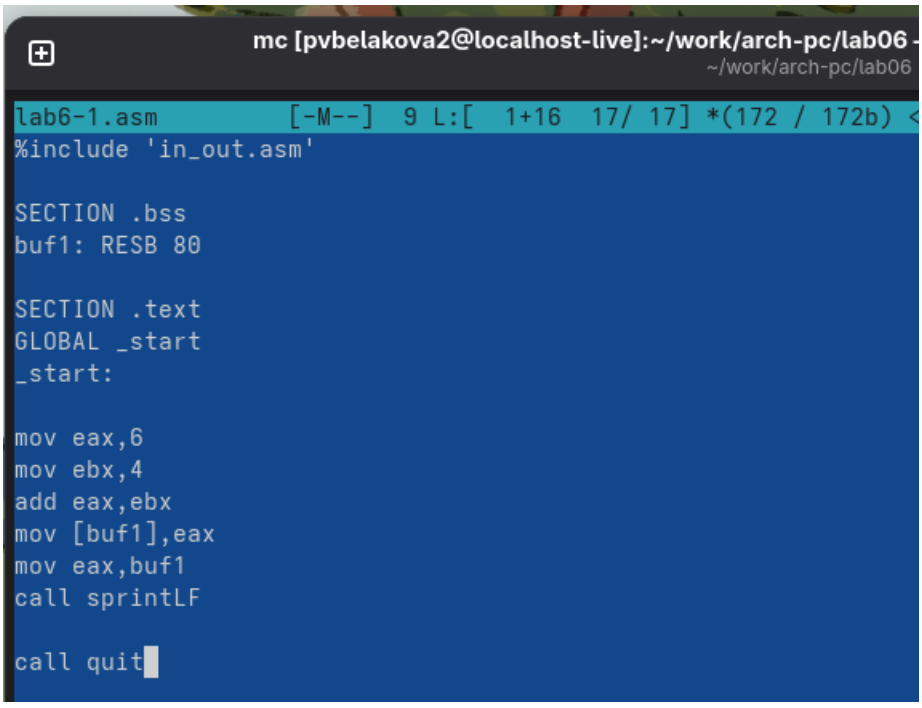

Рисунок 4 - измененный код программы.

Создаю исполняемый файл и запускаю его (рисунок 5).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm -o lab6-1_1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1_1 lab6-1_1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ./lab6-1_1


pvbelakova2@localhost-live:~/work/arch-pc/lab06$
```

Рисунок 5 - создание исполняемого файла и запуск программы.

В данном случае выводится символ с кодом 10 - перенос строки. Отображается ли этот символ при выводе на экран? Да, так как при завершении программы в консоли выведено две пустых строки.

4. Создаю файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и ввожу в него текст программы из листинга 6.2 (рисунок 6).

```
mc [pvbelakova2@localhost-live]:~/work/arch-pc/lab06
                                                    ~/work/arch-pc/lab06
lab6-2.asm          [-M--]  0 L:[  1+11  12/ 12] *(111 / 120b) 06
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF

call quit
```

Рисунок 6 - код программы.

Создаю исполняемый файл и запускаю его (рисунок 7).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ touch lab6-2.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ./lab6-2
106
pvbelakova2@localhost-live:~/work/arch-pc/lab06$
```

рисунок 7 - создание файла lab6-2.asm, создание объектного и исполняемого файлов и запуск программы.

В результате работы программы мы получили число 106.

5. Аналогично предыдущему примеру изменяю символы на числа (рисунок 8).

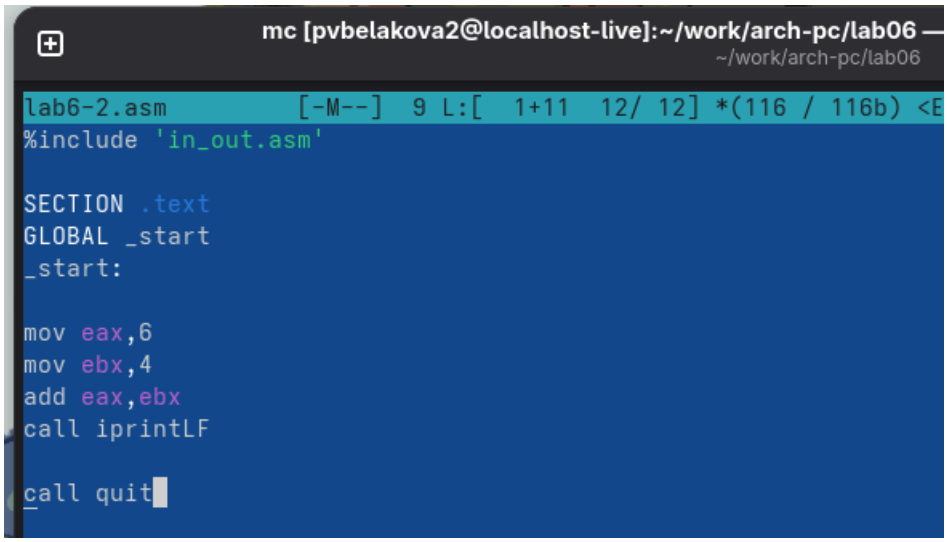

Рисунок 8 - код программы с функцией iprintLF.

Создаю исполняемый файл и запустите его (рисунок 9).

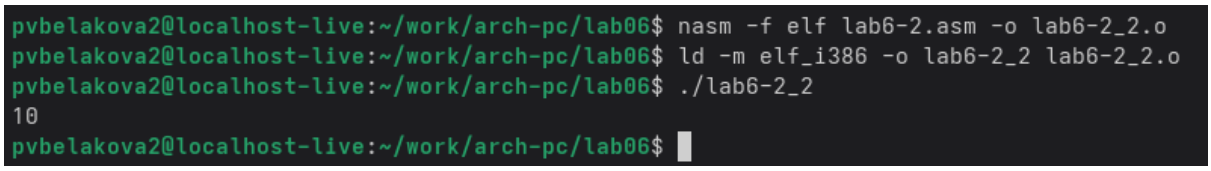

Рисунок 9 - создание объектного и исполняемого файлов и запуск программы.

При исполнении программы будет получено число 10.

Заменяю функцию iprintLF на iprint. (рисунок 10).

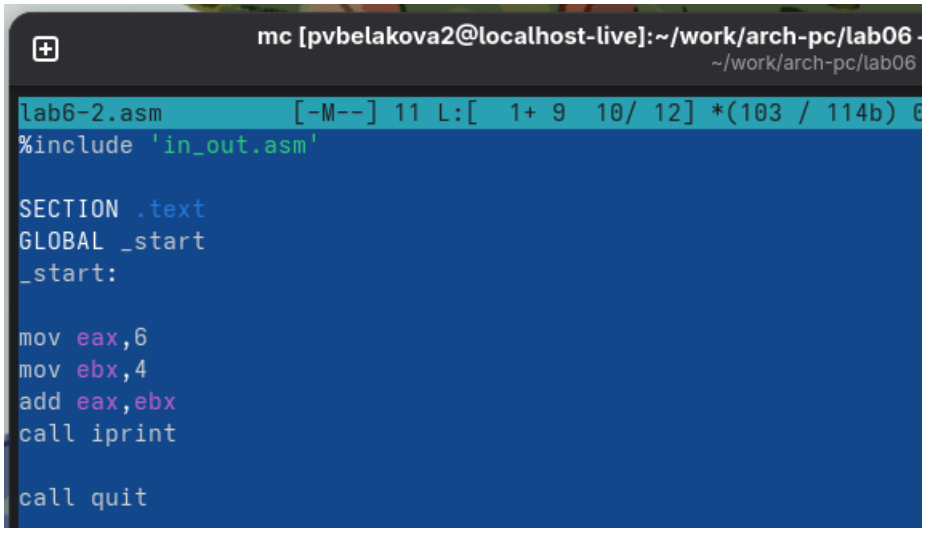

Рисунок 10 - код программы с функцией iprint.

Создаю исполняемый файл и запускаю его (рисунок 11).

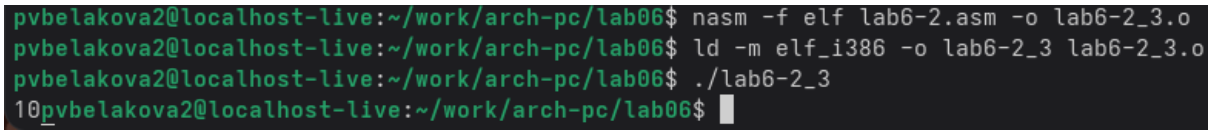

Рисунок 11 - создание объектного и исполняемого файлов и запуск программы.

Выводы функций iprintLF и iprint отличаются тем, что функция iprintLF после вывода делает перенос строки, а iprint не делает из-за чего следующий текст выводится на этой же строке.

Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm (рисунок 12) и ввожу в него текст программы (рисунок 13).

Рисунок 12 - создание файла lab6-3.asm.

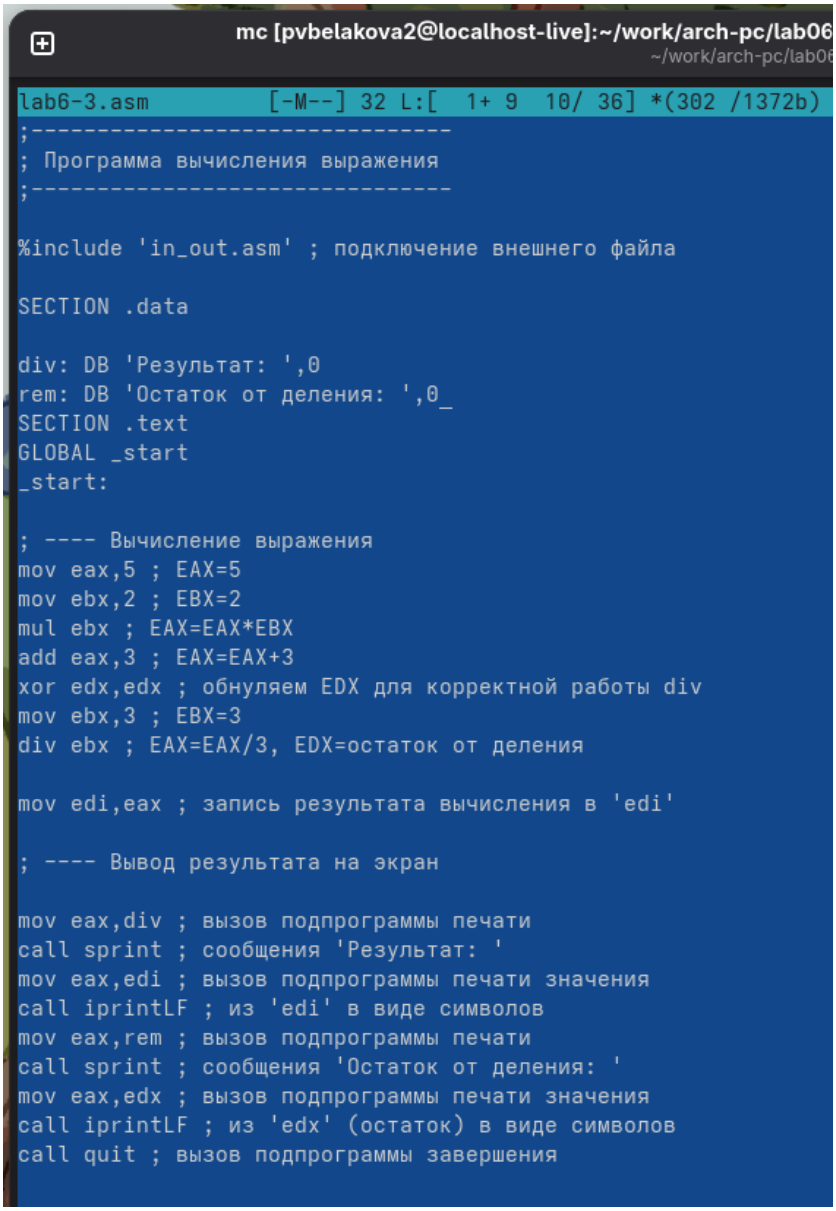

Рисунок 13 - код программы для вычисления функции $f(x) = (5 * 2 + 3)/3$.

Создаю исполняемый файл и запускаю его (рисунок 14).

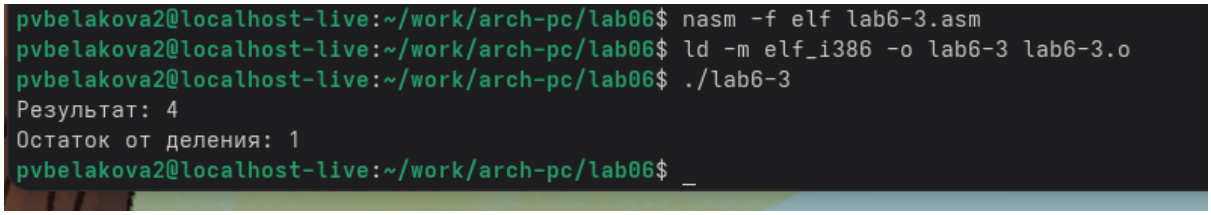

Рисунок 14 - создание объектного и исполняемого файлов и запуск программы.

Изменяю текст программы для вычисления выражения $f(x) = (4 * 6 + 2)/5$ (рисунок 15).

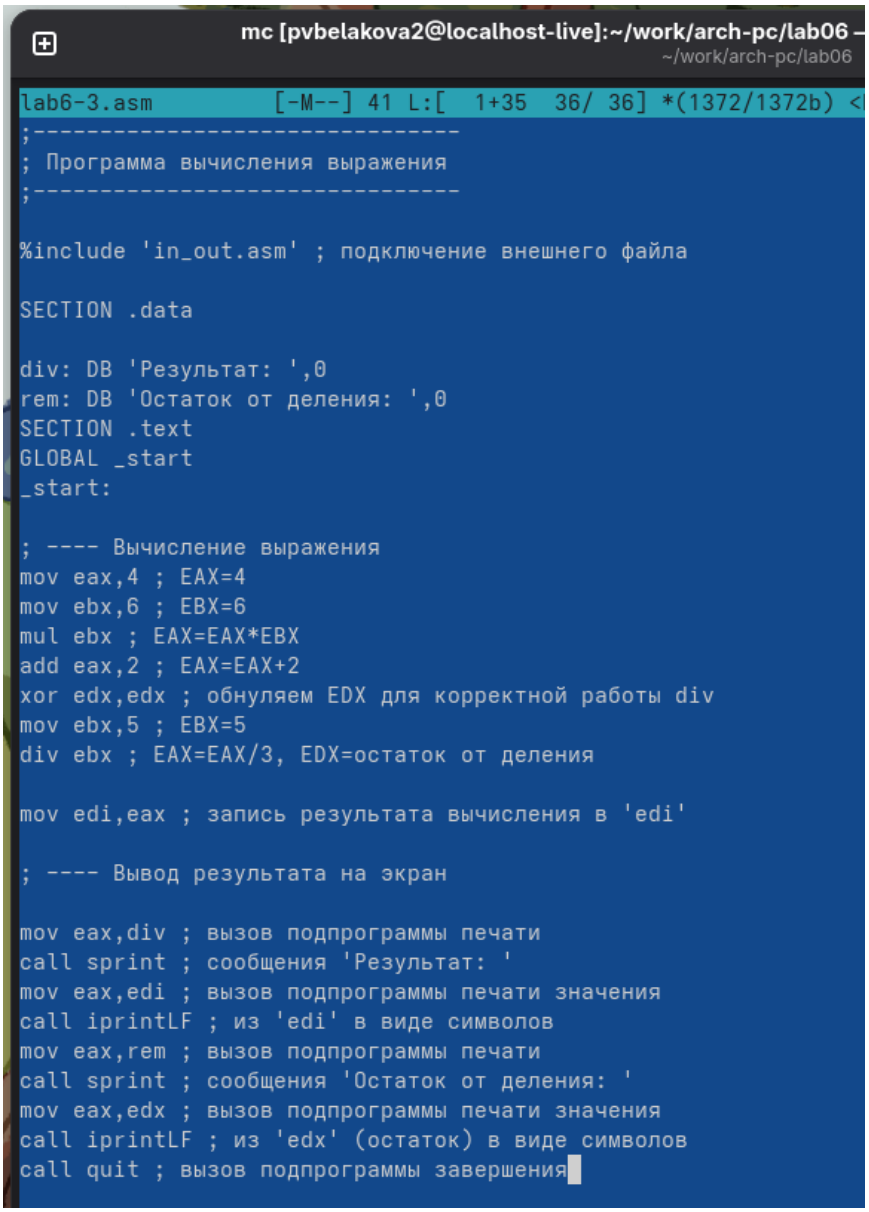

Рисунок 15 - код программы для вычисления функции $f(x) = (4 * 6 + 2)/5$.

Создаю исполняемый файл и проверяю его работу (рисунок 16).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm -o lab6-3_1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3_1 lab6-3_1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ./lab6-3_1
Результат: 5
Остаток от деления: 1
pvbelakova2@localhost-live:~/work/arch-pc/lab06$
```

Рисунок 16 - создание объектного и исполняемого файлов и запуск программы.

Создаю файл variant.asm в каталоге (рисунок 18), ввожу код файл variant.asm (рисунок 17).

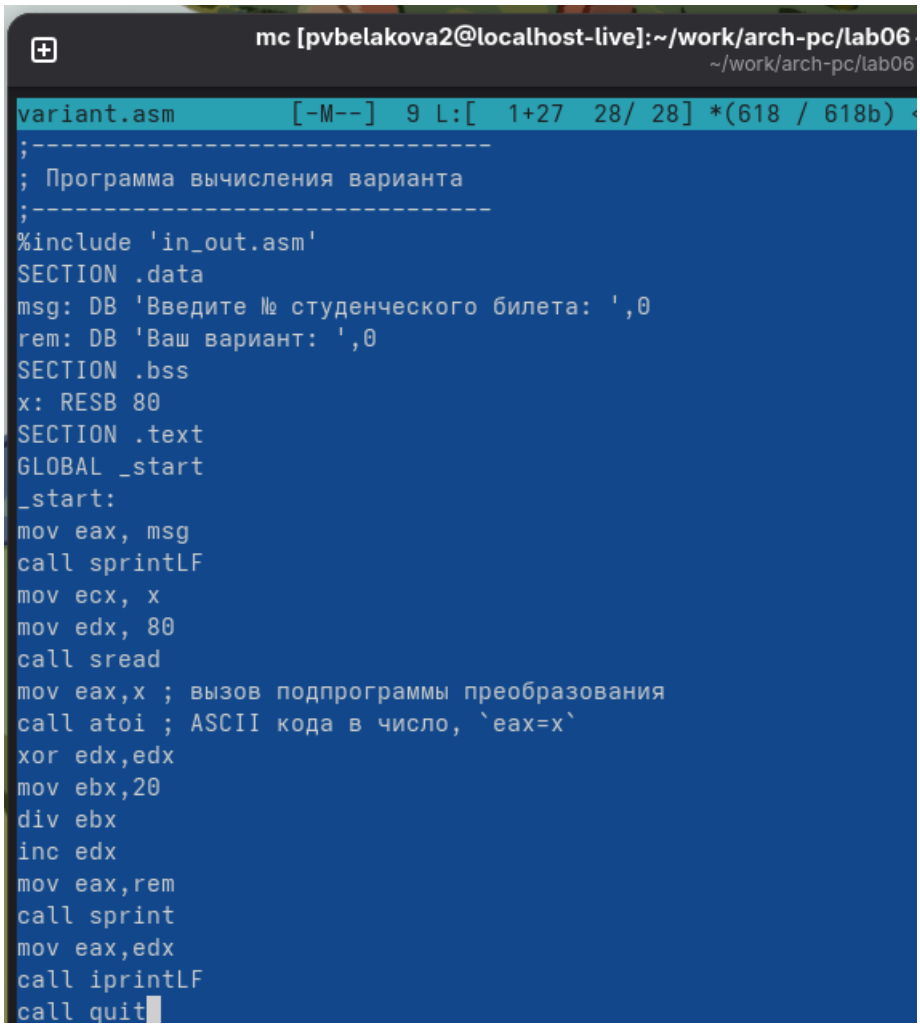

Рисунок 17 - код программы.

Создаю исполняемый файл и запускаю его (рисунок 18).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ touch variant.asm

pvbelakova2@localhost-live:~/work/arch-pc/lab06$ nasm -f elf variant.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
pvbelakova2@localhost-live:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032252589
Ваш вариант: 10
pvbelakova2@localhost-live:~/work/arch-pc/lab06$
```

Рисунок 18 - создание объектного и исполняемого файлов и запуск программы, вычисляющей вариант по номеру студенческого билета.

При аналитической проверке результата ответ получился - 10, что совпадает с решением программы.

1. За вывод на экран сообщения ‘Ваш вариант:’ отвечают строки: mov eax,rem и call sprint.
2. Инструкция mov ecx, x используется для помещения адреса буфера x в регистр ecx, mov edx, 80 для установки максимальной длины ввода, call sread для вызова функции чтения строки из stdin.
3. Инструкция “call atoi” преобразует ASCII-строку в число.
4. За вычисление варианта отвечают строки: xor edx,edx - обнуление edx для деления, mov ebx,20 - делитель = 20., div ebx - вычисляет eax = eax/20 и edx = остаток, inc edx - увеличение остатка на 1.
5. При выполнении инструкции “div ebx” остаток от деления записывается в регистр edx.
6. Инструкция “inc edx” увеличивает значение в регистре edx (остаток от деления) на 1.
7. За вывод на экран результата вычислений отвечают строки: mov eax,edx - помещает результат (вариант) в eax, call sprintLF - выводит число и перевод строки.

Задание для самостоятельной работы

1. Написать программу вычисления выражения $y = f(x)$ вариант 10 (рисунок 19).

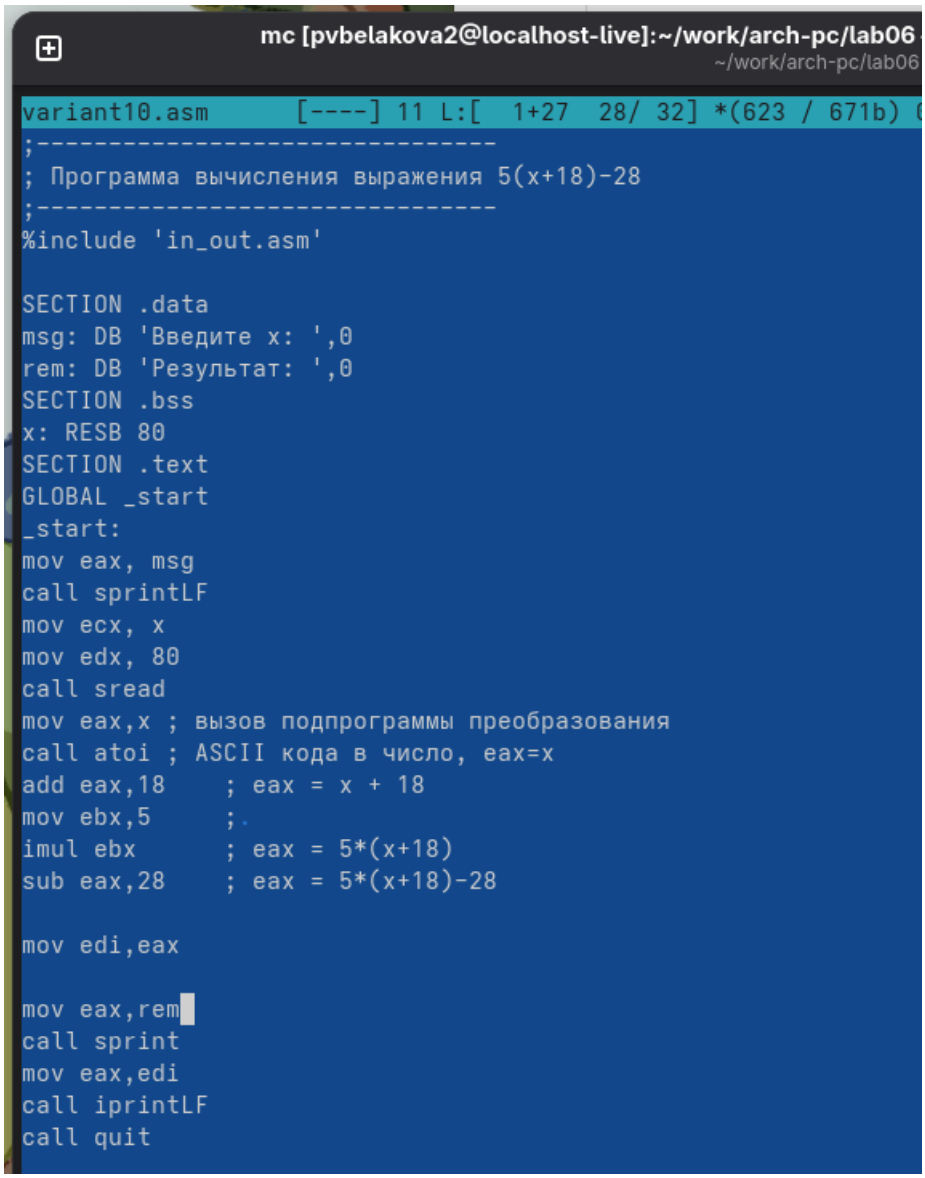

Рисунок 19 - код программы для вычисления функции 5(x+18)-20.

Создаю исполняемый файл и запускаю его (рисунок 20).

Рисунок 20 - создание объектного и исполняемого файлов и запуск программы.

Выводы

В ходе лабораторной работы были изучены арифметических инструкций языка ассемблера NASM.