**Отчет по лабораторной работе №6**

*дисциплина: Архитектура компьютера*

Сергеев Даниил Олегович

Содержание

# 1 Цель лабораторной работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Ход выполнения лабораторной работы

1. Создаем каталог ~/work/arch-pc/lab06, переходим в него и создаем lab6-1.asm. В файл вводим текст программы из листинга 6.1.

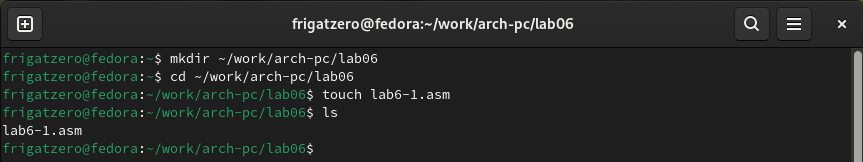


Рис. 1: Создание lab6-1.asm

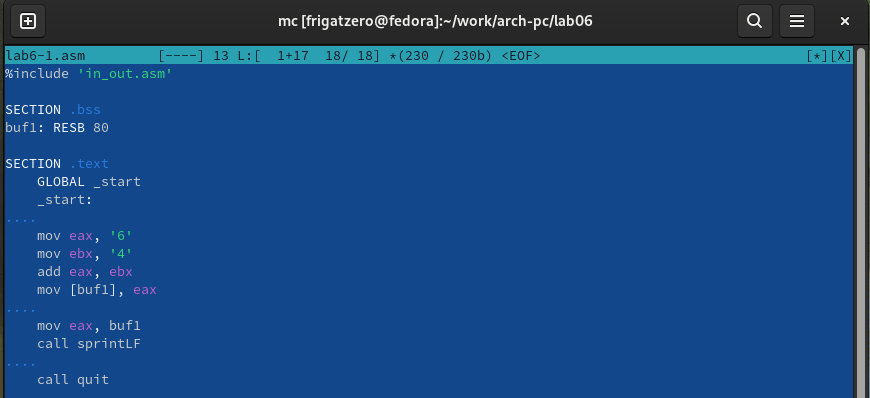


Рис. 2: Код из листинга 6.1

1. Создаем исполняемый файл и запускаем его.

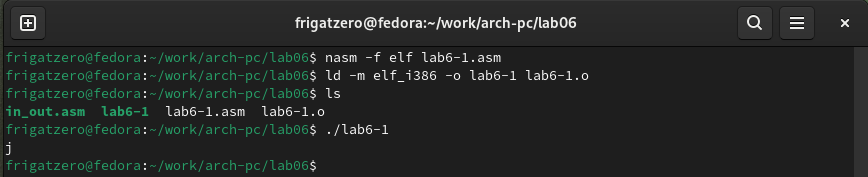


Рис. 3: Проверка работы файла lab6-1

1. Изменяем текст программы. Запишем в регистры числа вместо символов ‘4’ и ‘6’. Создаем исполняемый файл и запускаем его. В результате работы программы получаем код символа 10, который соответствует символу переноса строки, поэтому мы не видим его при выводе на экран.

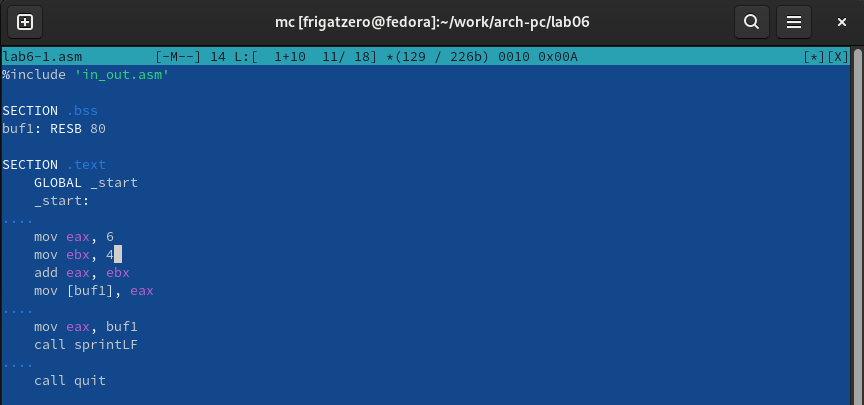


Рис. 4: Изменяем текст программы lab6-1.asm

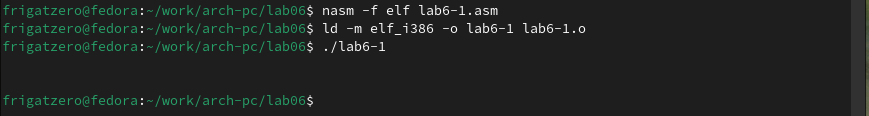


Рис. 5: Проверка работы измененного файла lab6-1

1. Создаем файл lab6-2.asm и вводим в него текст из листинга 6.2.

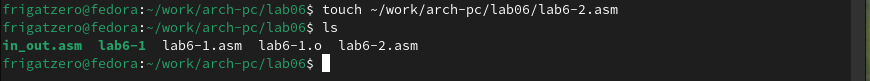


Рис. 6: Создание lab6-2.asm

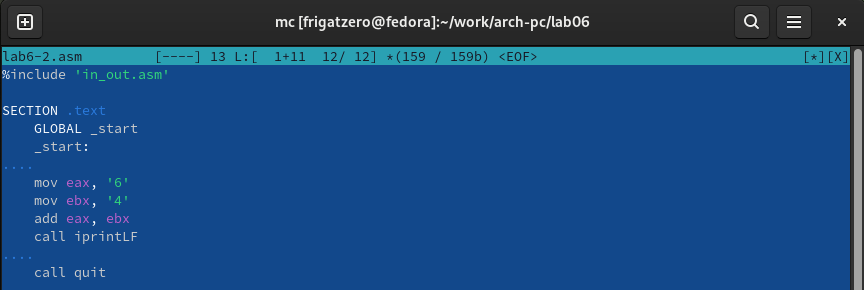


Рис. 7: Код из листинга 6.2

1. Создаем исполняемый файл lab6-2 и запускаем его.

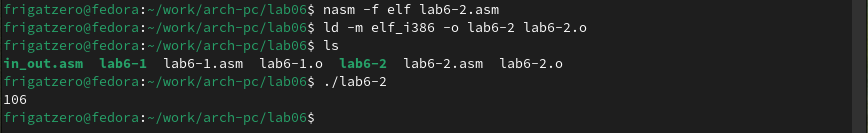


Рис. 8: Проверка работы файла lab6-2

1. Аналогично изменяем символы ‘4’ и ‘6’ на числа. Проверяем работу файла. При исполнении программы получаем результат равный 10.

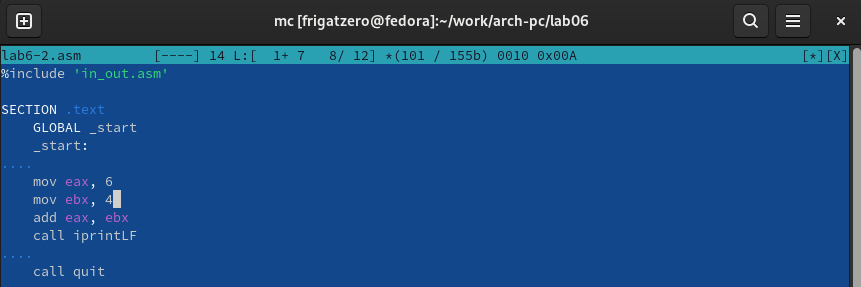


Рис. 9: Изменяем текст программы lab6-2.asm

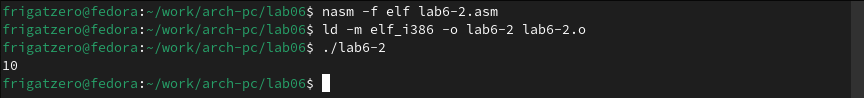


Рис. 10: Проверка работы измененного файла lab6-2

1. Заменяем функцию iprintLF на iprint. Создаем исполняемый файл и запускаем его. При исполнении программы также получаем 10, но, так как функция iprint не ставит символ переноса строки, сразу после вывода программы начинается строка ввода.

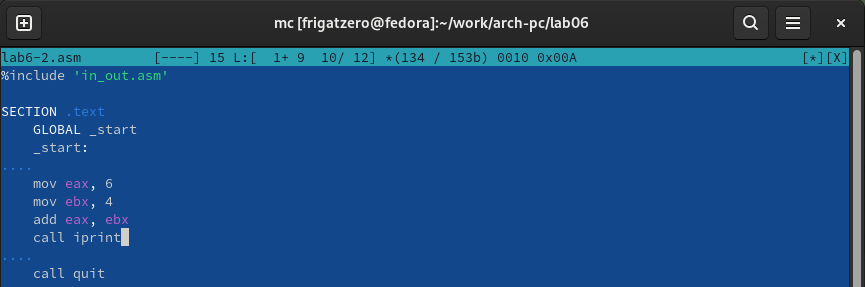


Рис. 11: Изменяем iprintLF на iprint

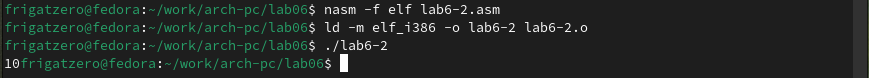


Рис. 12: Проверка работы lab6-2

1. Создаем файл lab6-3.asm и вводим в него текст из листинга 6.3.

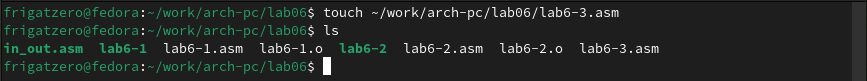


Рис. 13: Создание lab6-3.asm

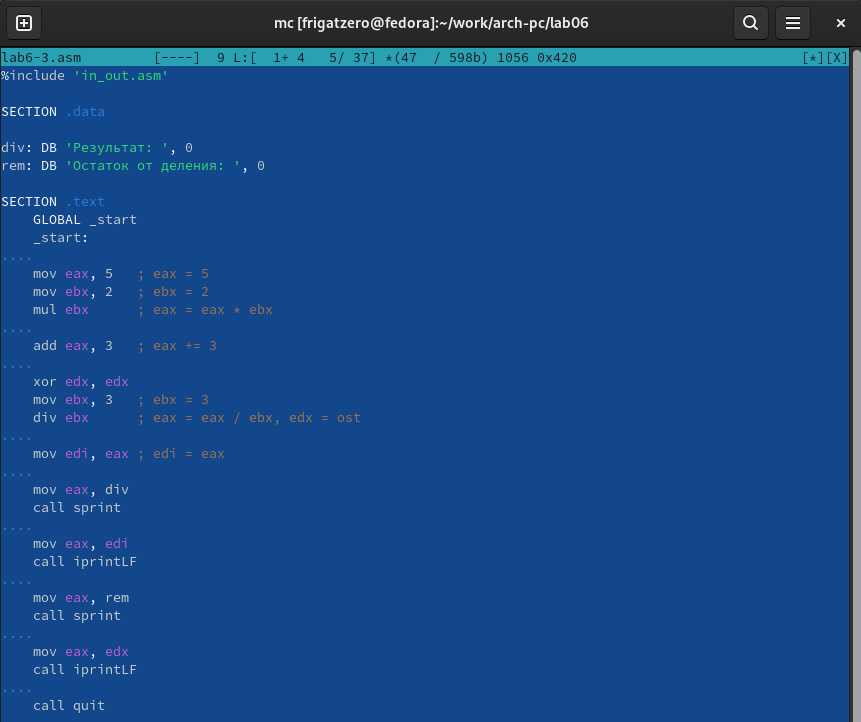


Рис. 14: Код из листинга 6.3

1. Создаем исполняемый файл lab6-3 и запускаем его.

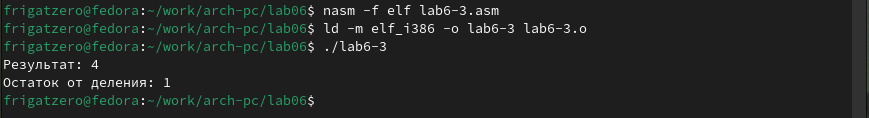


Рис. 15: Проверка работы файла lab6-3

1. Изменяем текст программы для вычисления выражения f(x) = (4\*6 + 2)/5. Создаем исполняемый файл и проверяем его работу.

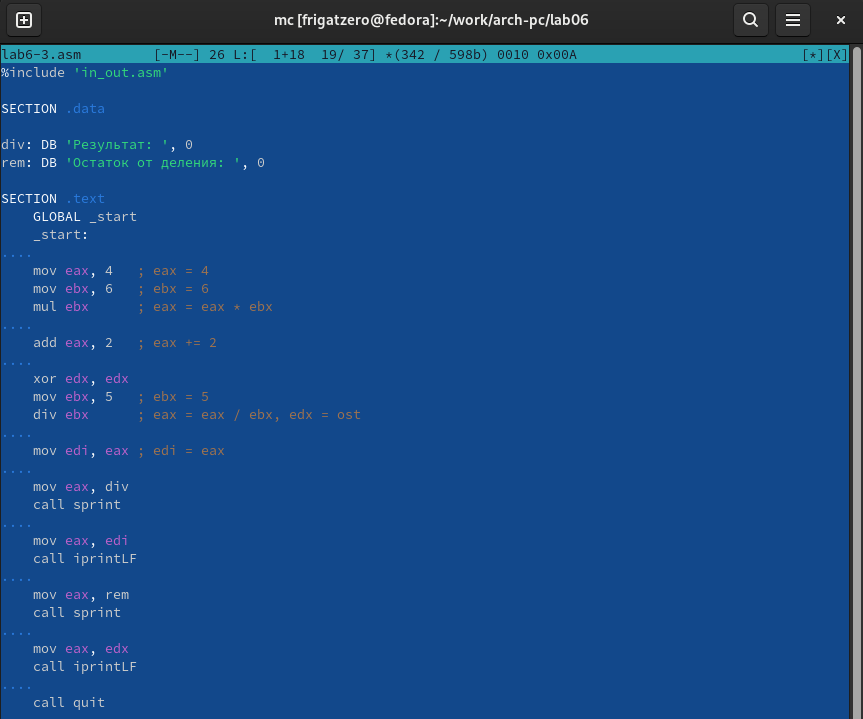


Рис. 16: Изменяем файл под выражение f(x) = (4\*6 + 2)/5

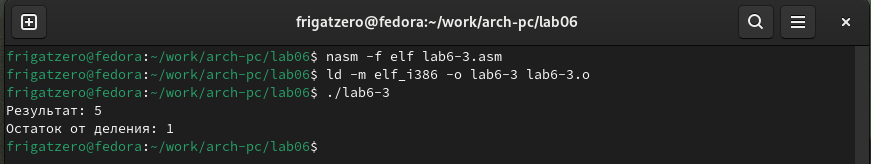


Рис. 17: Проверка работы измененного файла lab6-3

1. Создаем файл variant.asm и вводим в него текст из листинга 6.4.

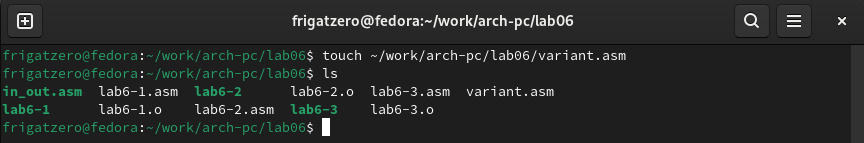


Рис. 18: Создание variant.asm

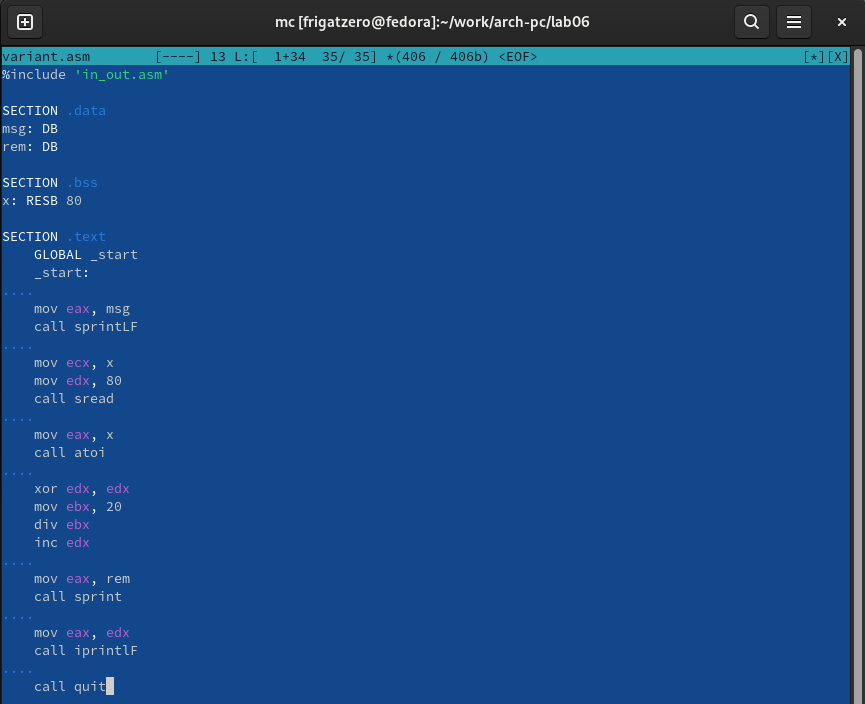


Рис. 19: Код из листинга 6.4

1. Создаем исполняемый файл, проверяем его. Вычисляем номер варианта задания для самостоятельной работы. Результат работы программы и аналитическое решение совпали.

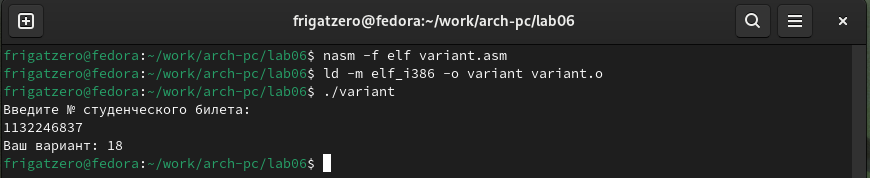


Рис. 20: Проверка работы variant.asm

1. Ответы на вопросы:
   1. За вывод сообщения “Ваш вариант:” на экран отвечают строки:

mov eax,rem  
call sprint

* 1. Данные инструкции используются для записи введенной строки в переменную x.
  2. Инструкция call atoi используется для вызова функции, которая преобразовывает ascii-код символа в целое число.
  3. За вычисление варианта отвечают строки:

xor edx, edx  
mov ebx, 20  
div ebx  
inc edx

* 1. Остаток от деления при выполнении инструкции div ebx записывается в регистр edx.
  2. Инструкция inc edx используется для увеличения значения указанного регистра на единицу.
  3. За вывод на экран результата вычислений отвечают строки:

mov eax, edx  
call iprintLF

# 3 Ход выполнения заданий для самостоятельной работы

1. Создаем программу для вычисления выражения №18 из таблицы 6.3. Для этого создаем файл work.asm и в окне mc пишем код программы

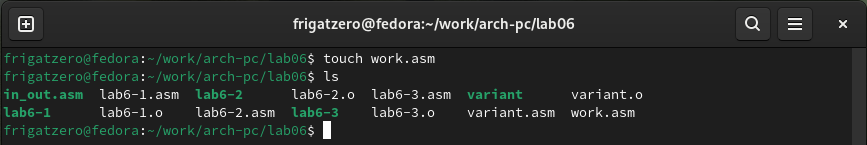


Рис. 21: Создаем файл work.asm

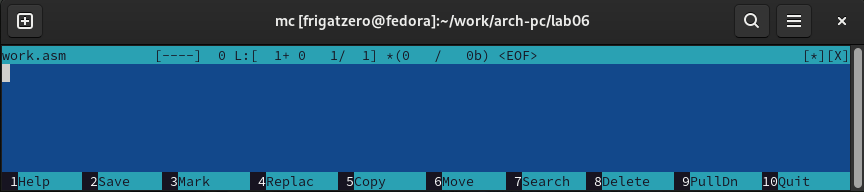


Рис. 22: Окно редактора Midnight Commander

**Листинг 4.1** Программа для вычисления выражения y = f(x).

%include 'in\_out.asm'  
  
SECTION .data  
exp: DB 'Программа выводит значение выражения №18 f(x) = 3(x+10)-20. ',0  
inp: DB 'Введите x: ',0  
ans: DB 'Результат вычислений: ',0  
  
SECTION .bss  
x: RESB 80  
  
SECTION .text  
 GLOBAL \_start  
 \_start:  
   
 mov eax, exp ; Вывод сообщения exp  
 call sprintLF ;  
   
 mov eax, inp ; Вывод сообщения inp  
 call sprint ;  
   
 mov ecx, x ; Записываем символы  
 mov edx, 80 ; из ввода в переменную  
 call sread ; (буфер) x  
   
 mov eax, x ; Переводим значение x из  
 call atoi ; кода ascii в число и  
 ; и записываем его в eax   
   
 add eax, 10 ; x + 10  
   
 mov ebx, 3 ;   
 mul ebx ; 3 \* (x + 10)  
   
 sub eax, 20 ; 3 \* (x + 10) - 20  
   
 mov edi, eax ; Сохраняем результат в edi  
   
 mov eax, ans ; Вывод сообщения ans  
 call sprint ;  
   
 mov eax, edi ; Вывод результата сразу  
 call iprintLF ; после ans  
   
 call quit

1. Проверяем работу программы для x равного 1 и 5.

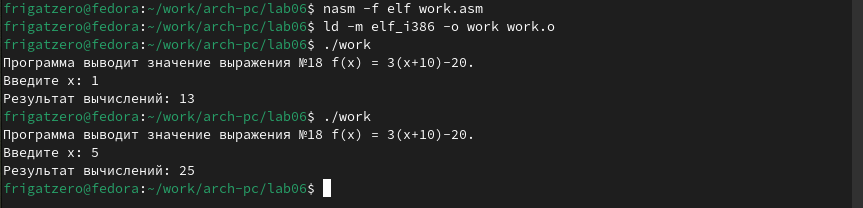


Рис. 23: Проверка работы задания для самостоятельной работы

# 4 Вывод

После выполнения заданий лабораторной работы и заданий для самостоятельной работы я освоил арифмитические инструкции add, sub, mul, div, inc и dec, приобрел опыт в написании собственных программ для вычисления выражений на языке ассемблера NASM.