Шаблон отчёта по лабораторной работе

Лабораторная работа №6.Арифметические операции в NASM.

Акрур Имад НКАбд-06-24

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Описание результатов выполнения лабораторной работы:

## 2.1 Символьные и численные данные в NASM

1. Переходим в созданную директорию:

cd ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab06/report

1. Создаем файл для программы:

touch lab6-1.asm

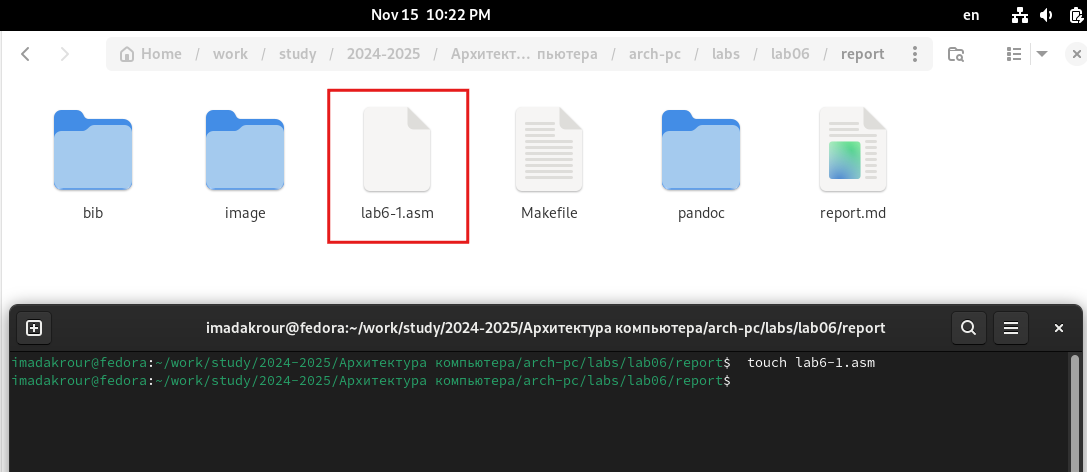


Рис. 1: Создание файла программы lab6-1.asm

Команда touch создаёт пустой файл с указанным именем, который позже будет заполнен программным кодом.

### 2.1.1 Написание программы для вывода значений регистров

Необходимо написать программу, которая выводит значения из регистров eax и ebx после выполнения операций с ними. Используется NASM и подключаемый файл in\_out.asm для упрощения работы с вводом и выводом.

1. Открываем файл lab6-1.asm для редактирования:

gedit lab6-1.asm]

1. Вводим текст программы:

%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

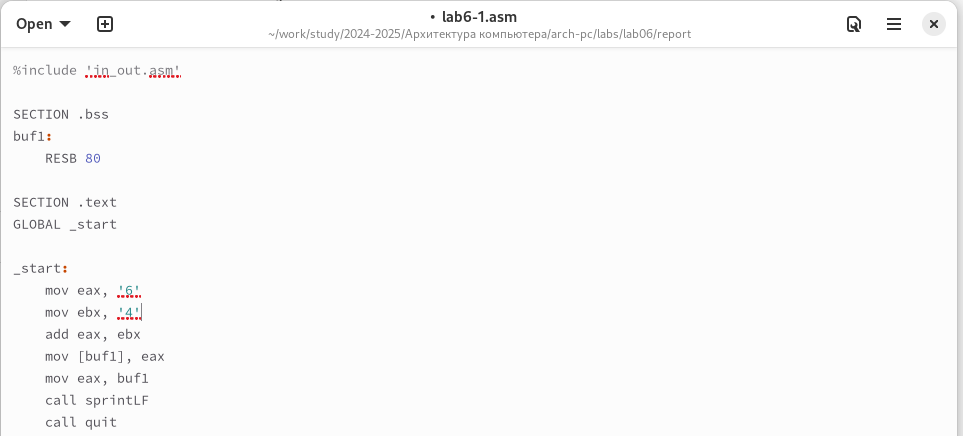


Рис. 2: Открытие файла lab6-1.asm в текстовом редакторе

1. Сохраняем файл и выходим из редактора
2. Создание исполняемого файла

* Компилируем программу:

nasm -f elf lab6-1.asm

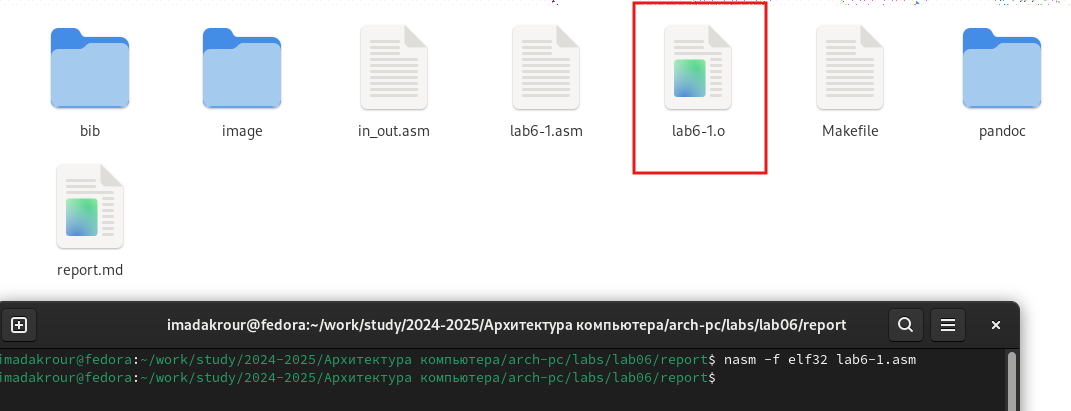


Рис. 3: Компиляция программы lab6-1.asm

* Линкуем с использованием ld:

ld -m elf\_i386 -o lab6-1 lab6-1.o

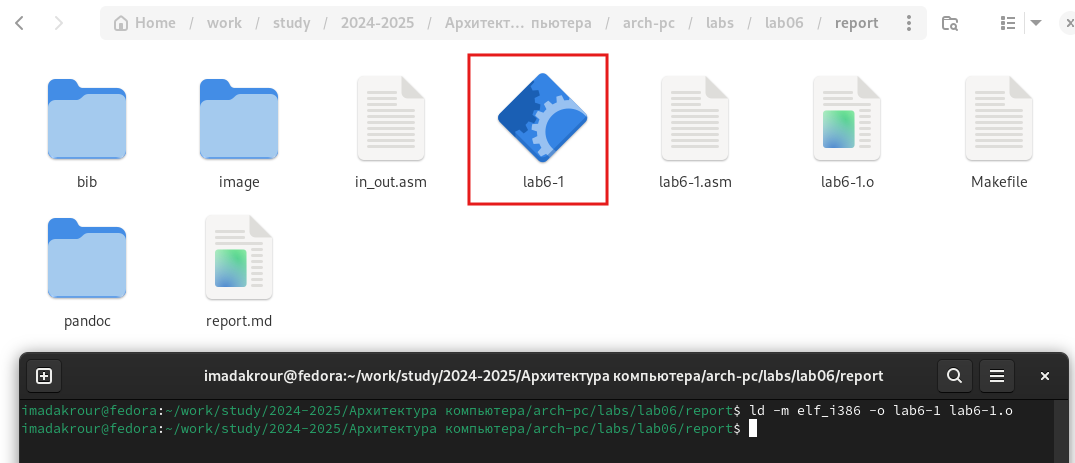


Рис. 4: Линковка программы с помощью команды ld

* Запускаем программу:

./lab6-1

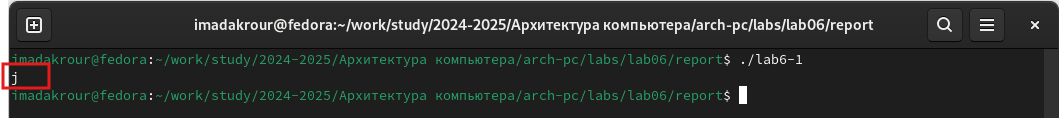


Рис. 5: Запуск программы и вывод результата (символ ‘j’)

На этом этапе была написана и успешно выполнена программа, которая складывает ASCII-коды символов '6' и '4'. Результатом сложения стал символ с кодом 106 (символ 'j'), что подтверждает работу арифметической операции в контексте кодов символов.

#### 2.1.1.1 Изменение программы для работы с числами

Необходимо изменить программу так, чтобы вместо символов использовались числа. В регистры eax и ebx записываются числа 6 и 4, их сумма вычисляется, и результат выводится.

1. Открываем файл lab6-1.asm для редактирования:

gedit lab6-1.asm

1. Изменяем строки:

mov eax, 6  
mov ebx, 4

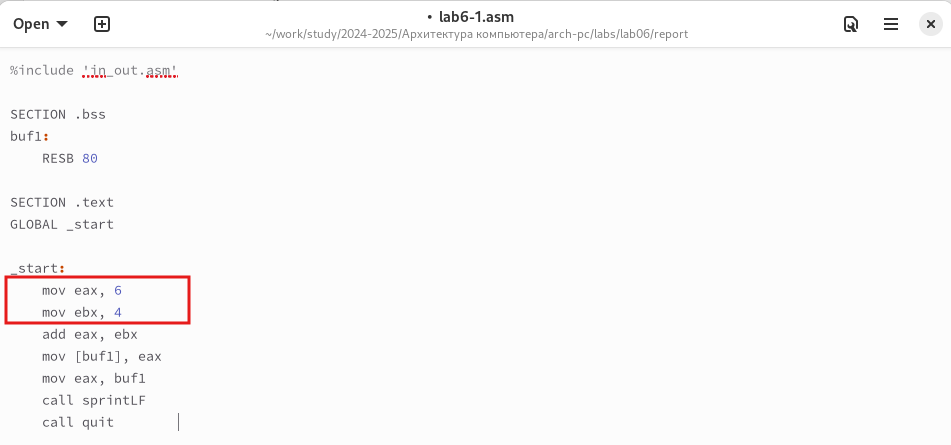


Рис. 6: Изменение программы для работы с числами вместо символов

1. Повторяем компиляцию и выполнение:

nasm -f elf lab6-1.asm  
ld -m elf\_i386 -o lab6-1 lab6-1.o  
./lab6-1

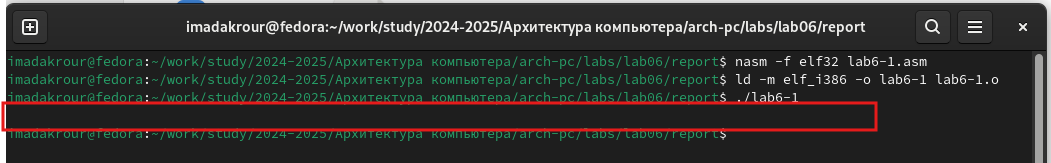


Рис. 7: Запуск программы с числами и вывод результата

Изменение кода позволило работать с числами, а не с символами. Однако при выводе результатом стал символ с кодом 10 (новая строка). Это связано с тем, что в регистре содержится ASCII-код числа, а не само число.

### 2.1.2 Выполнение задания по преобразованию текста программы и анализу результатов

#### 2.1.2.1 Создание файла и ввод текста программы

Создайте файл lab6-2.asm:

touch ~/work/arch-pc/lab06/lab6-2.asm



Рис. 8: Создание файла программы lab6-2.asm

следующий код в файл lab6-2.asm:

%include 'in\_out.asm'  
  
SECTION .text  
GLOBAL \_start  
  
\_start:  
 mov eax, '6' ; Загрузка ASCII-кода символа '6' в регистр EAX  
 mov ebx, '4' ; Загрузка ASCII-кода символа '4' в регистр EBX  
 add eax, ebx ; Сложение кодов символов  
 call iprintLF ; Вызов функции для вывода результата с переводом строки  
 call quit ; Завершение программы

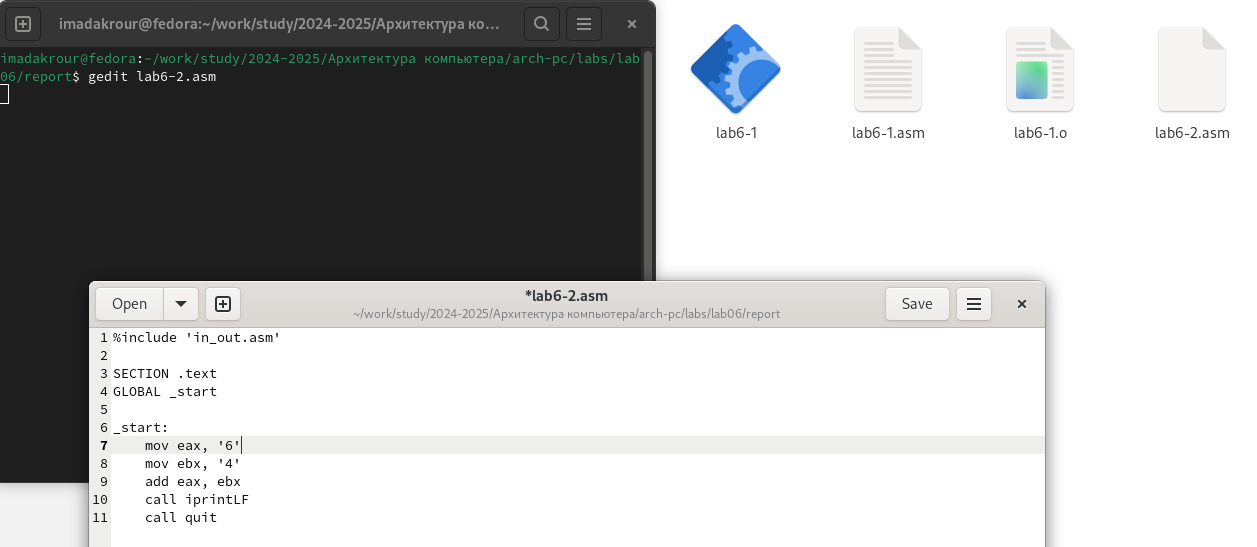


Рис. 9: Ввод текста программы для сложения ASCII-кодов в lab6-2.asm

#### 2.1.2.2 Компиляция и запуск программы

Выполните следующие команды для компиляции и запуска программы:

nasm -f elf lab6-2.asm  
ld -m elf\_i386 -o lab6-2 lab6-2.o  
./lab6-2

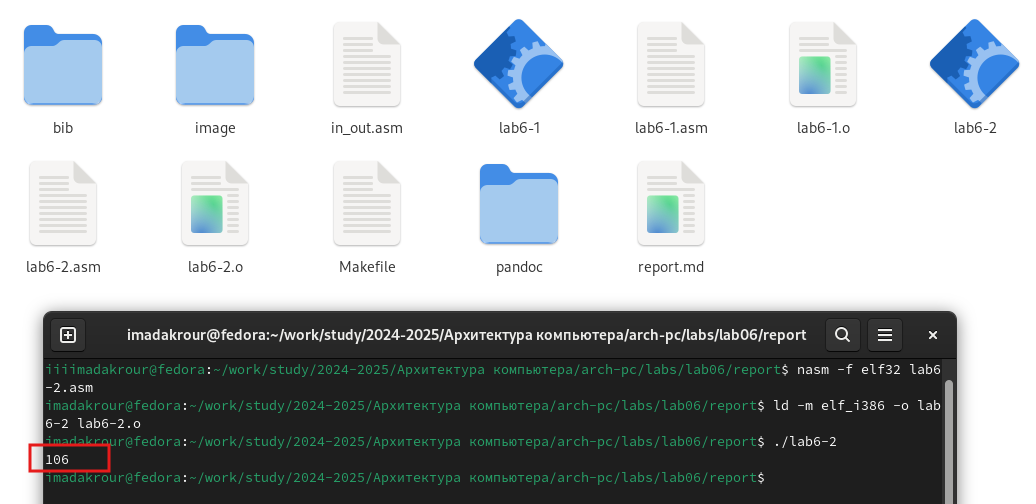


Рис. 10: Запуск программы lab6-2.asm и вывод результата (106)

**Результат выполнения:**

Программа выведет число **106**, так как происходит сложение ASCII-кодов символов '6' и '4' (54 + 52 = 106). Функция iprintLF позволяет вывести результат сложения как число.

#### 2.1.2.3 Изменение символов на числа

Замените строки:

mov eax, '6'  
mov ebx, '4'

на:

mov eax, 6  
mov ebx, 4

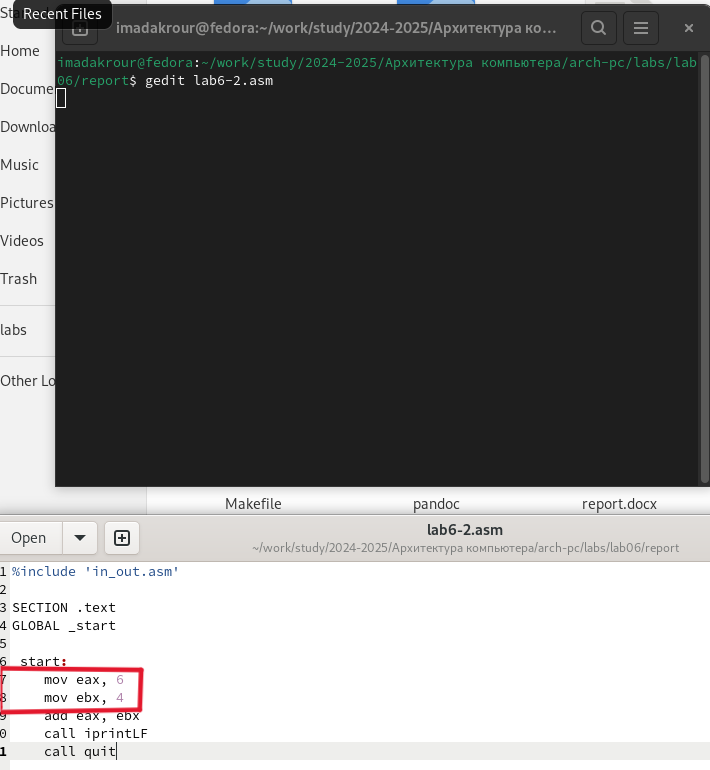


Рис. 11: зменение символов на числа

Обновлённый код:

%include 'in\_out.asm'  
  
SECTION .text  
GLOBAL \_start  
  
\_start:  
 mov eax, 6 ; Загрузка числа 6 в регистр EAX  
 mov ebx, 4 ; Загрузка числа 4 в регистр EBX  
 add eax, ebx ; Сложение чисел  
 call iprintLF ; Вызов функции для вывода результата с переводом строки  
 call quit ; Завершение программы

#### 2.1.2.4 Компиляция и запуск обновлённой программы

nasm -f elf lab6-2.asm  
ld -m elf\_i386 -o lab6-2 lab6-2.o  
./lab6-2

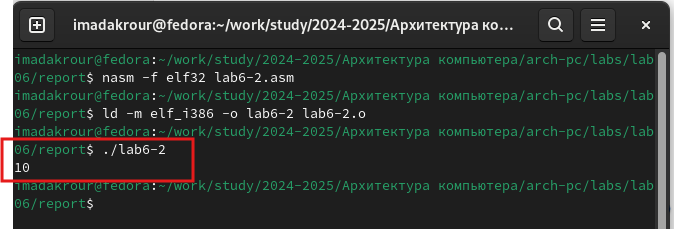


Рис. 12: Компиляция и запуск обновлённой программы

**Результат выполнения:**

Программа выведет **10**, так как теперь складываются числа, а не ASCII-коды. Это разница между использованием символов и чисел.

#### 2.1.2.5 Замена iprintLF на iprint

Замените строку:

call iprintLF

на:

call iprint

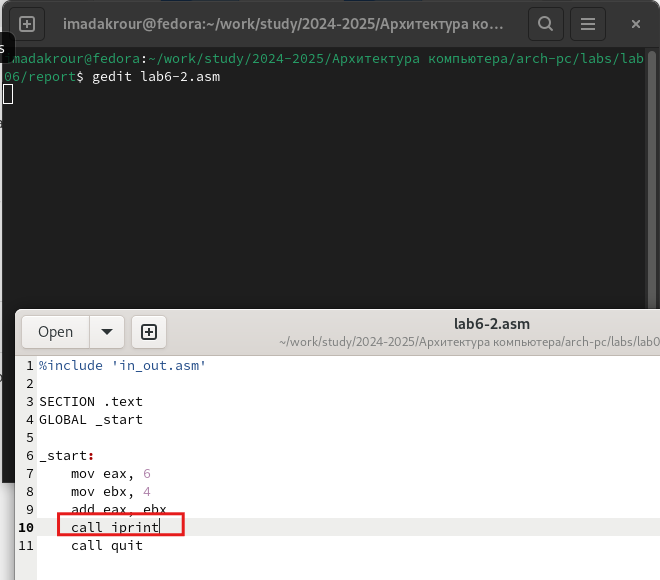


Рис. 13: Изменение функции вывода на iprint

#### 2.1.2.6 Компиляция и запуск программы с iprint

nasm -f elf lab6-2.asm  
ld -m elf\_i386 -o lab6-2 lab6-2.o  
./lab6-2

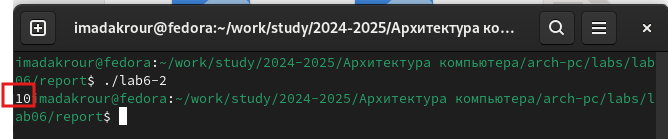


Рис. 14: Запуск программы с iprint и вывод результата на той же строке.)

**Различие между iprintLF и iprint:**

* iprintLF выводит результат и переводит курсор на новую строку.
* iprint просто выводит результат, оставаясь на той же строке. В данном случае результат будет **10**, но курсор останется в конце числа.

### 2.1.3 Выполнение арифметических операций в NASM

Написать программу, вычисляющую результат выражения ( f(x) = (5 \* 2 + 3)/3 ). Результат и остаток от деления выводятся на экран.

1. Создаем файл lab6-3.asm:

touch lab6-3.asm

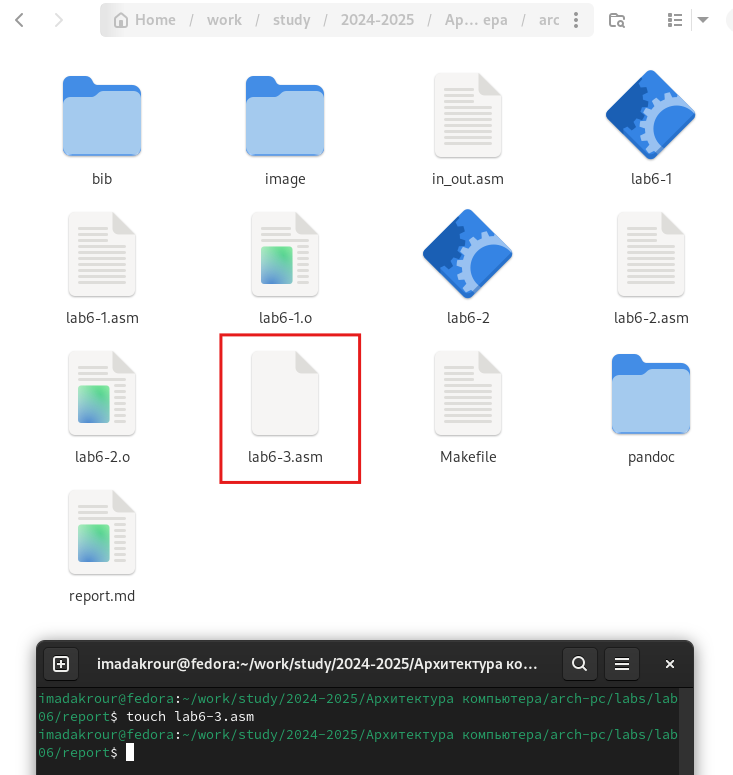


Рис. 15: Создание программы для вычисления выражения f(x)= (5⋅2+3)/3

1. Вводим текст программы:

%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

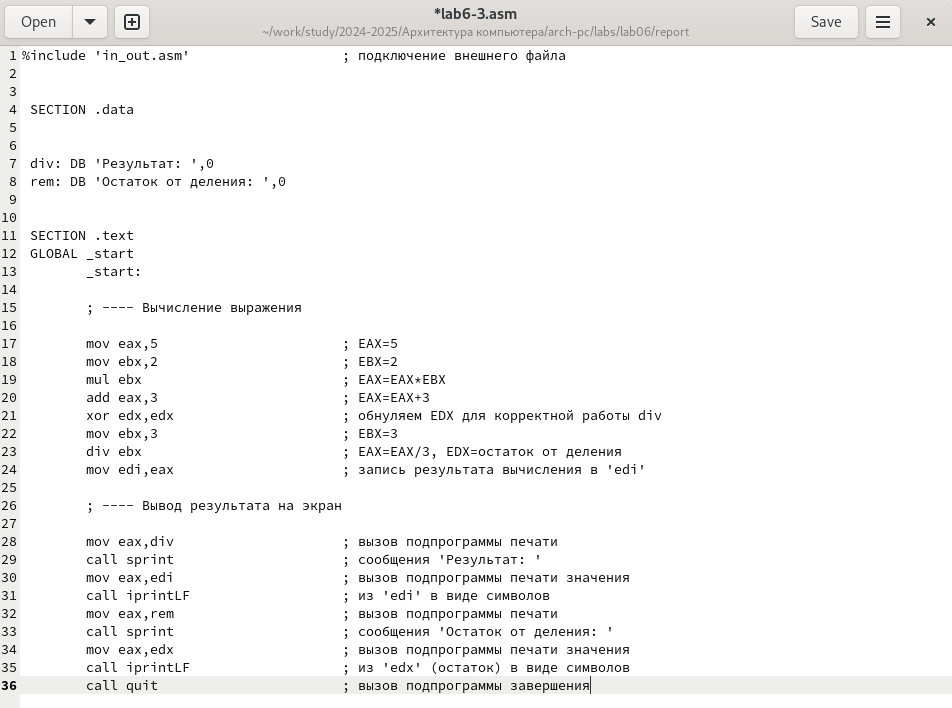


Рис. 16: Ввод текста программы

1. Компилируем и запускаем:

nasm -f elf lab6-3.asm  
ld -m elf\_i386 -o lab6-3 lab6-3.o  
./lab6-3

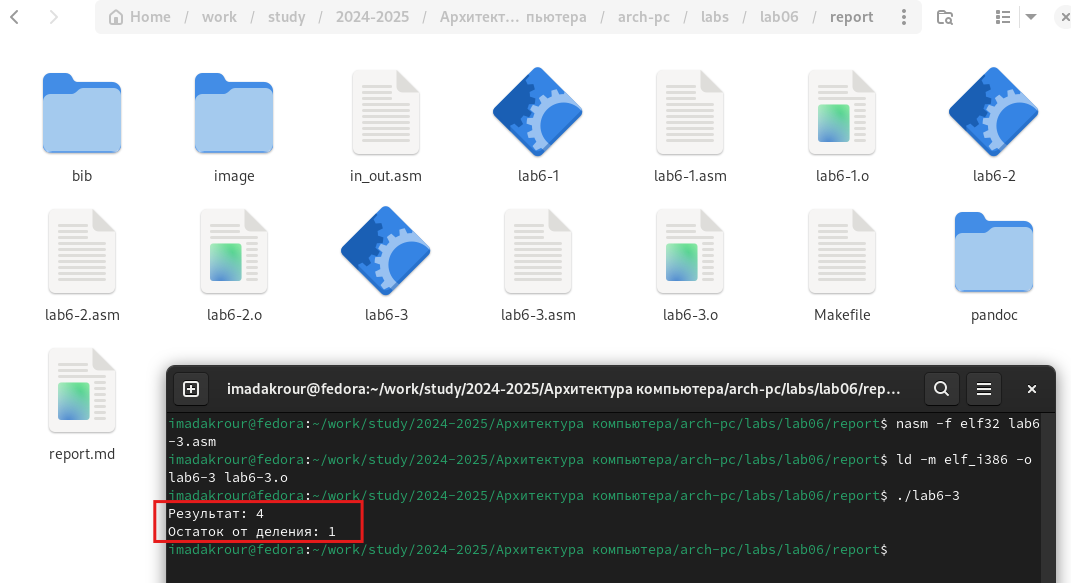


Рис. 17: Результат выполнения (Результат: 4, Остаток от деления: 1)

Программа успешно вычисляет заданное выражение. Остаток от деления и целая часть результата отображаются корректно.

Измените текст программы для вычисления выражения 𝑓(𝑥) = (4 ∗ 6 + 2)/5. Создайте исполняемый файлипроверьте его работу :

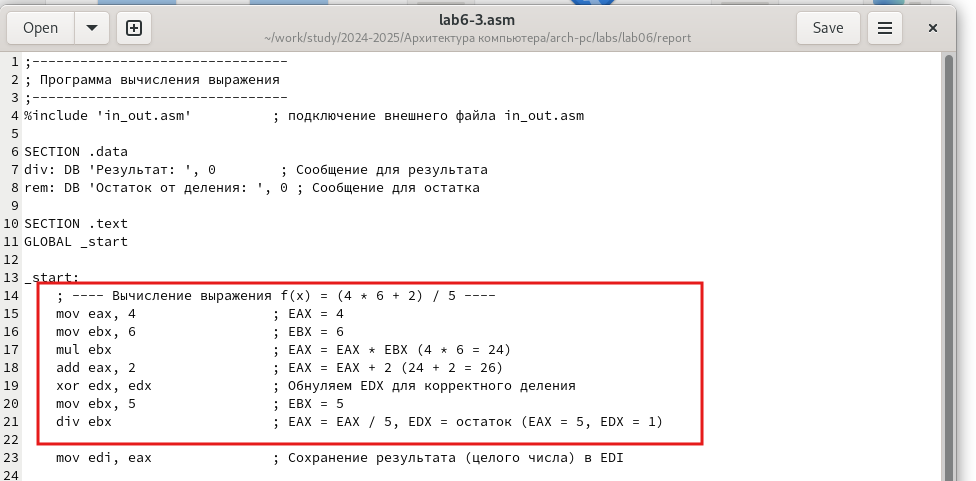


Рис. 18: Ввод текста программы

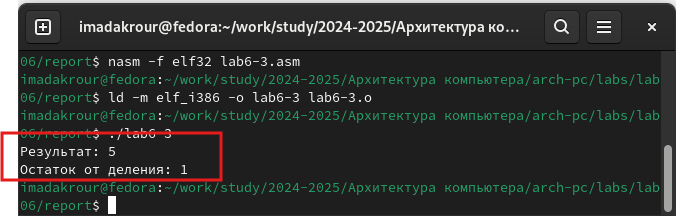


Рис. 19: Результат выполнения (Результат: 5, Остаток от деления: 1)

1. Создание программы для вычисления варианта задания по номеру студенческого билета

Необходимо написать программу, которая запрашивает номер студенческого билета, вычисляет номер варианта по формуле ( = () + 1 ), и выводит его.

* Создаем файл variant.asm:

touch ~/work/arch-pc/lab06/variant.asm

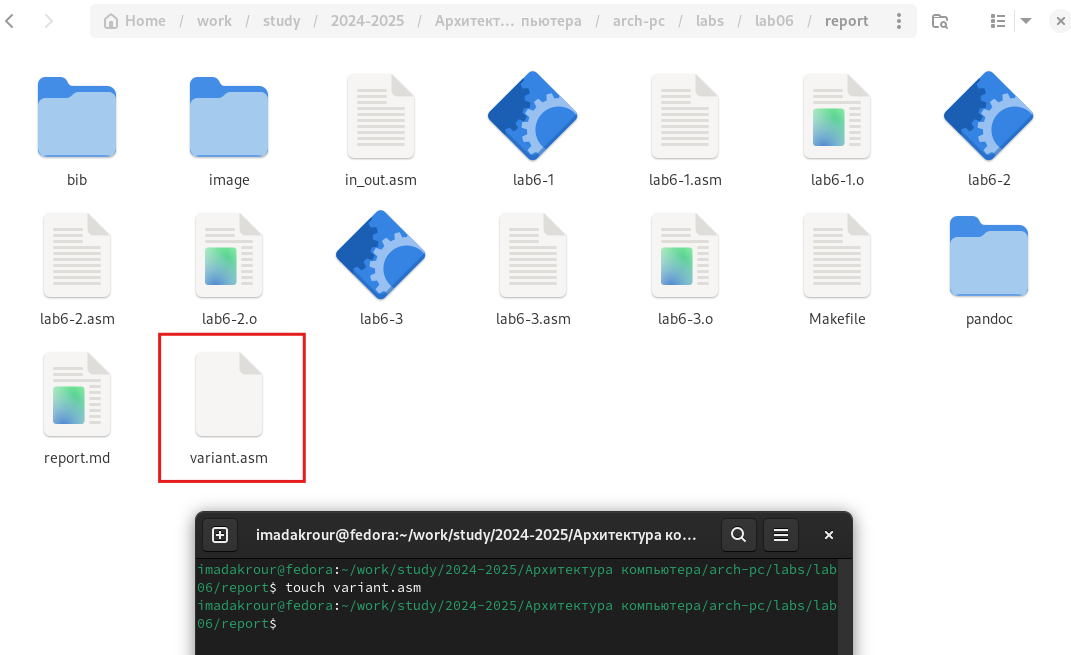


Рис. 20: Создаем файл variant.asm

* Вводим текст программы:

;--------------------------------  
; Программа вычисления варианта  
;--------------------------------  
%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 ; вывод сообщения  
 call sprintLF  
 mov ecx, x ; указатель на буфер для ввода  
 mov edx, 80 ; длина ввода  
 call sread ; ввод номера студенческого билета  
 mov eax, x ; преобразование из ASCII в число  
 call atoi  
 xor edx, edx ; обнуление регистра edx  
 mov ebx, 20 ; делитель для операции mod  
 div ebx ; деление, результат в eax, остаток в edx  
 inc edx ; +1 к остатку  
 mov eax, rem ; сообщение 'Ваш вариант: '  
 call sprint  
 mov eax, edx ; вывод остатка (результата)  
 call iprintLF  
 call quit ; завершение программы

* Компилируем и запускаем программу:

nasm -f elf variant.asm  
ld -m elf\_i386 -o variant variant.o  
./variant

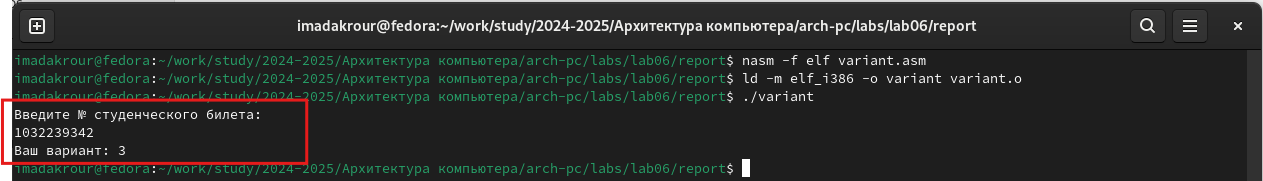


Рис. 21: Результат выполнения программы (Ваш вариант: 3)

Программа корректно принимает номер студенческого билета, рассчитывает номер варианта, используя остаток от деления на 20, и добавляет единицу. Это позволяет точно определить вариант, связанный с номером студента.

### 2.1.4 Ответы на вопросы по листингу 6.4

#### 2.1.4.1 Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

За вывод сообщения “Ваш вариант:” отвечают следующие строки:

mov eax, rem ; Загрузка адреса строки "Ваш вариант:" в регистр eax  
call sprint ; Вызов функции для вывода строки на экран

#### 2.1.4.2 Для чего используются следующие инструкции?

mov ecx, x ; Загрузка адреса буфера для ввода данных в регистр ecx  
mov edx, 80 ; Указание максимального размера вводимой строки (80 байт)  
call sread ; Вызов функции для считывания пользовательского ввода

**Назначение:**

Эти инструкции используются для подготовки к чтению данных с клавиатуры. - mov ecx, x: устанавливает адрес, куда будет записан ввод. - mov edx, 80: указывает максимальное количество символов, которое можно ввести. - call sread: инициирует процесс ввода данных пользователем.

#### 2.1.4.3 Для чего используется инструкция call atoi?

Инструкция call atoi используется для преобразования строки, содержащей символы ASCII, введённой пользователем, в числовое значение. После выполнения этой команды числовое значение вводимого числа будет сохранено в регистре eax.

#### 2.1.4.4 Какие строки листинга 6.4 отвечают за вычисления варианта?

За вычисления варианта по формуле ((Sn mod 20) + 1) отвечают следующие строки:

xor edx, edx ; Обнуление регистра EDX для корректной работы операции деления  
mov ebx, 20 ; Установка делителя (20) в регистр EBX  
div ebx ; Деление значения в EAX на EBX, результат в EAX, остаток в EDX  
inc edx ; Увеличение остатка на 1

#### 2.1.4.5 В какой регистр записывается остаток от деления при выполнении инструкции div ebx?

Остаток от деления записывается в регистр EDX.

#### 2.1.4.6 Для чего используется инструкция inc edx?

Инструкция inc edx увеличивает значение в регистре EDX на 1. Это используется для добавления 1 к остатку от деления, что соответствует формуле вычисления варианта ((Sn mod 20) + 1).

#### 2.1.4.7 Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?

За вывод результата на экран отвечают следующие строки:

mov eax, edx ; Перенос результата (варианта) из регистра EDX в EAX  
call iprintLF ; Вывод результата в виде числа на экран с переводом строки

### 2.1.5 выводы по результатам выполнения заданий :

В ходе выполнения лабораторной работы были изучены базовые арифметические операции на языке ассемблера NASM, такие как сложение, умножение, деление и работа с остатком. Также реализованы программы для работы с данными в ASCII-формате и числами, включая использование функций из библиотеки in\_out.asm.

# 3 Описание результатов выполнения заданийдля самостоятельной работы:

## 3.1 описание выполняемого задания :

1. **Создаем файл программы**:

touch ~/work/arch-pc/lab06/test.asm

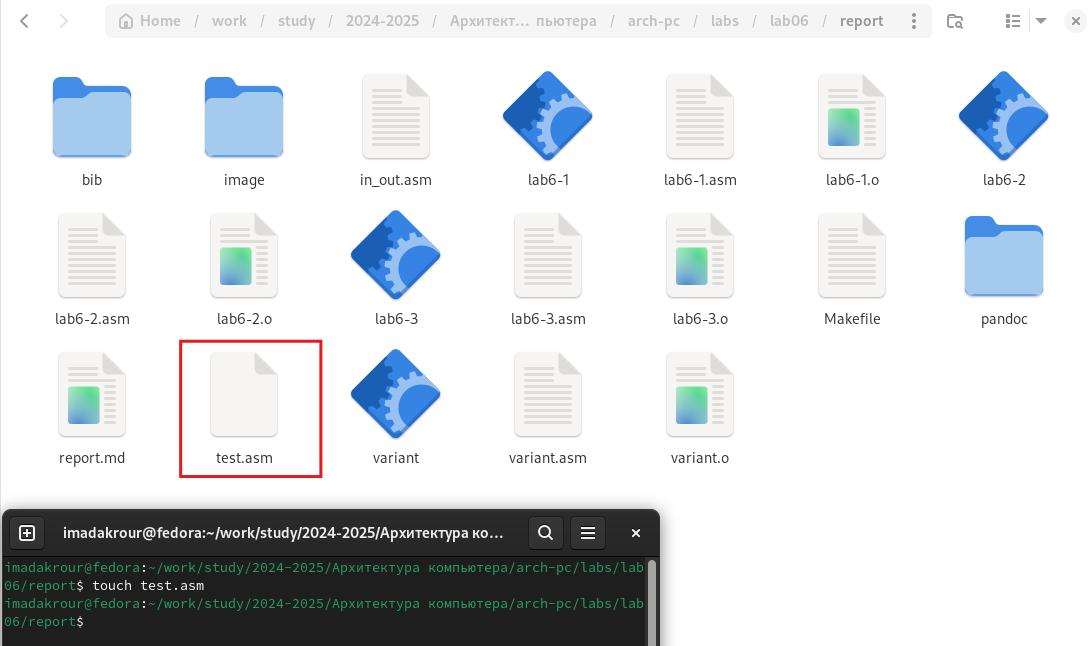


Рис. 22: Создание программы для вычисления функции

1. **Текст программы**:

;-------------------------------------  
; Программа вычисления функции f(x)  
; Вариант 3: f(x) = (2 + x)^2  
;-------------------------------------  
%include 'in\_out.asm'  
SECTION .data  
expr: DB 'f(x) = (2 + x)^2', 0  
prompt1: DB 'Введите x1: ', 0  
prompt2: DB 'Введите x2: ', 0  
result1: DB 'Результат f(x1): ', 0  
result2: DB 'Результат f(x2): ', 0  
SECTION .bss  
buf: RESB 80  
SECTION .text  
GLOBAL \_start  
\_start:  
 ; Вывод выражения  
 mov eax, expr  
 call sprintLF  
  
 ; Запрос ввода x1  
 mov eax, prompt1  
 call sprintLF  
 mov ecx, buf  
 mov edx, 80  
 call sread ; Ввод значения x1  
 mov eax, buf  
 call atoi ; Преобразование ASCII в число (x1 в eax)  
  
 ; Вычисление f(x1) = (2 + x1)^2  
 add eax, 2 ; eax = 2 + x1  
 mov ebx, eax ; Сохранение (2 + x1) в ebx  
 mul ebx ; eax = eax \* ebx = (2 + x1)^2  
 mov edi, eax ; Сохранение результата f(x1) в edi  
  
 ; Вывод результата f(x1)  
 mov eax, result1  
 call sprint  
 mov eax, edi  
 call iprintLF  
  
 ; Запрос ввода x2  
 mov eax, prompt2  
 call sprintLF  
 mov ecx, buf  
 mov edx, 80  
 call sread ; Ввод значения x2  
 mov eax, buf  
 call atoi ; Преобразование ASCII в число (x2 в eax)  
  
 ; Вычисление f(x2) = (2 + x2)^2  
 add eax, 2 ; eax = 2 + x2  
 mov ebx, eax ; Сохранение (2 + x2) в ebx  
 mul ebx ; eax = eax \* ebx = (2 + x2)^2  
 mov edi, eax ; Сохранение результата f(x2) в edi  
  
 ; Вывод результата f(x2)  
 mov eax, result2  
 call sprint  
 mov eax, edi  
 call iprintLF  
  
 ; Завершение программы  
 call quit

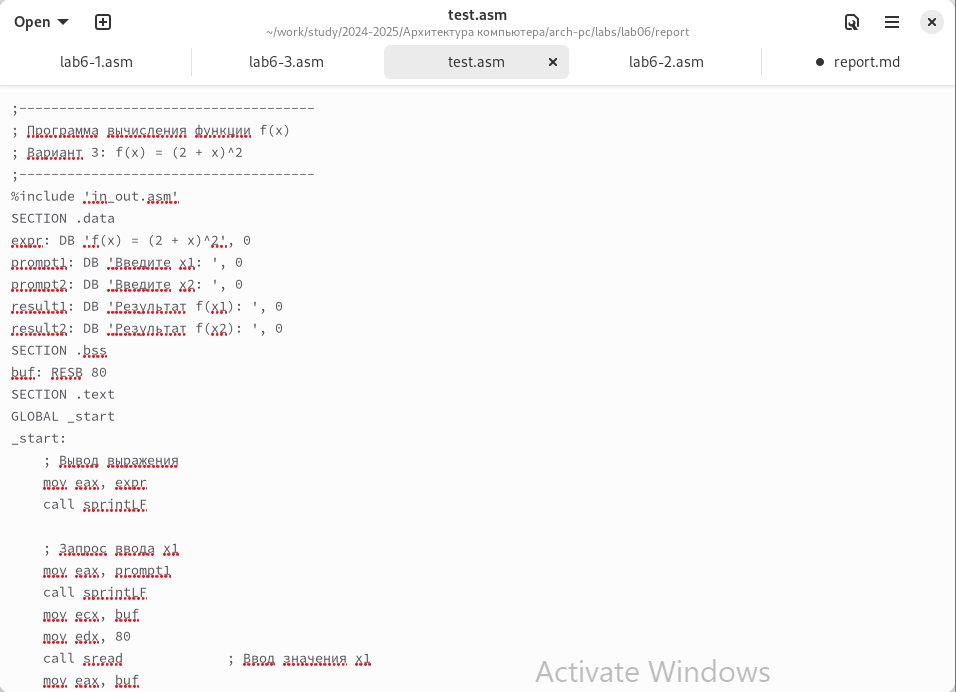


Рис. 23: Текст программы

1. Компилируем и запускаем программу:

nasm -f elf test.asm  
ld -m elf\_i386 -o test test.o  
./test

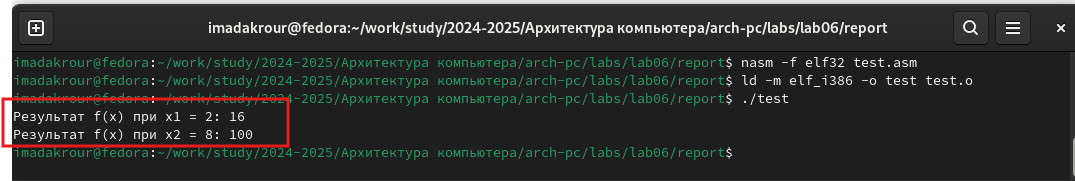


Рис. 24: Вывод результата

Программа корректно выполняет вычисления выражения ( f(x) = (2 + x)^2 ). В ходе работы были проверены значения ( x\_1 = 2 ) и ( x\_2 = 8 ). Результаты ( f(x) ) для обоих значений совпадают с аналитически рассчитанными: ( f(2) = 16 ) и ( f(8) = 100 ).

## 3.2 выводы по результатам выполнения заданий :

При выполнении самостоятельной работы реализована программа для вычисления функции f(x)=(2+x)^2. Программа запрашивает два значения x1 и x2, корректно вычисляет результат для каждого из них и выводит на экран. Проверены значения x1 = 2 и x2 = 8 результаты совпадают с аналитическими расчётами: f(2)=16 и f(8)=100. Работа показала важность использования арифметических операций в NASM, таких как сложение, возведение в квадрат и корректное преобразование данных. Все этапы программы, включая ввод, обработку и вывод данных, реализованы успешно.

# 4 Выводы

Лабораторная работа продемонстрировала, как с помощью языка ассемблера NASM можно выполнять сложные вычисления и работать с данными различного формата. Были освоены ключевые арифметические инструкции, работа с регистрами процессора и использование внешних функций для ввода и вывода данных. Все задания, включая лабораторные и самостоятельные, выполнены корректно, а полученные результаты соответствуют аналитическим расчётам.