Отчет по лабораторной работе №9

Архитектура компьютера

Никифоров Захар Сергеевич

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

2 Порядок выполнения работы

2.1 Реализация подпрограмм в NASM

Создаем каталог *lab09*, а в нем создаем файл *lab9-1.asm*, записываем код из *Листинг 8.1* и компилируем его.

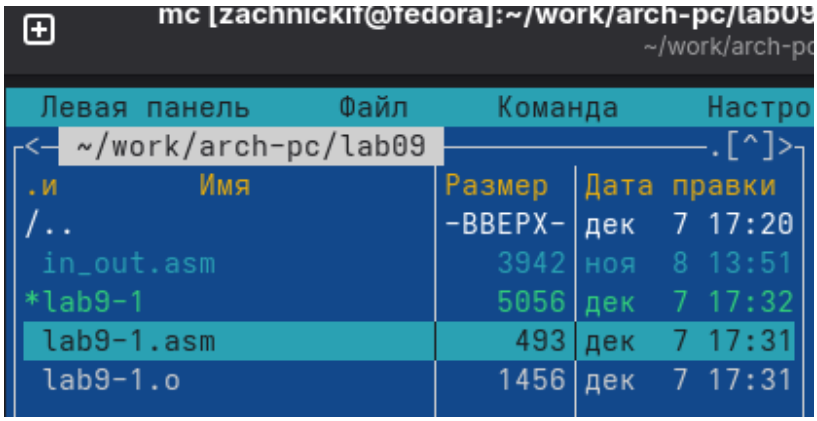

Рисунок 2.1: Скриншот терминала mc 1

Рисунок 2.2: Скриншот терминала mc 2

Получаем вывод результата функции. Запишем в код подпрограмму *_subcalcul*, вычисляющую f(g(x)), снова соберем и проверим вывод.

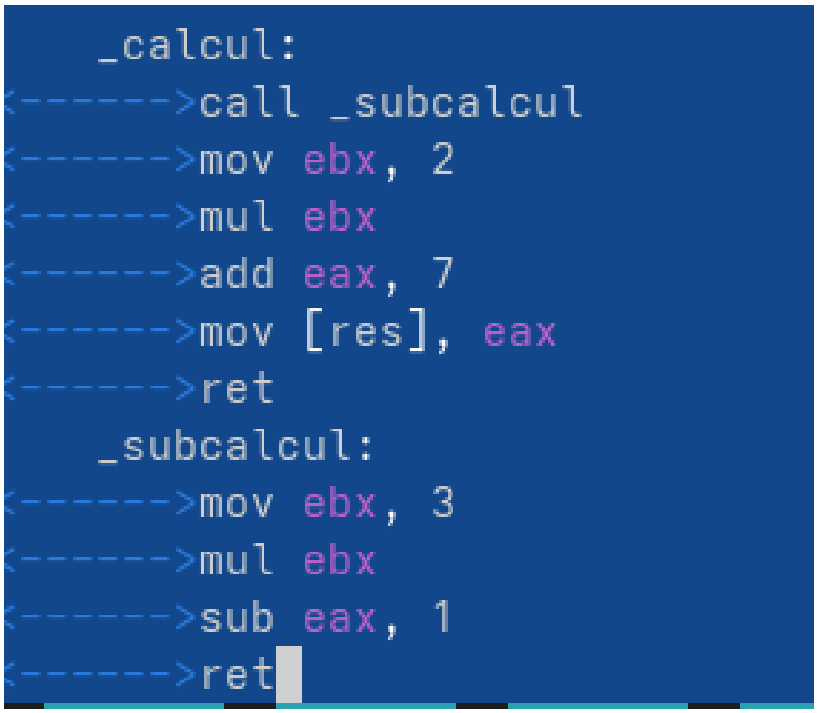

Рисунок 2.3: Скриншот терминала mc 3

Рисунок 2.4: Скриншот терминала mc 4

Мы видим, что программа работает корректно.

2.2 Отладка программам с помощью GDB

Создадим файл *lab9-2.asm*, запишем в него текст из *Листинг 9.2*, скомпилируем с получением файла трансляции и запустим в gdb.

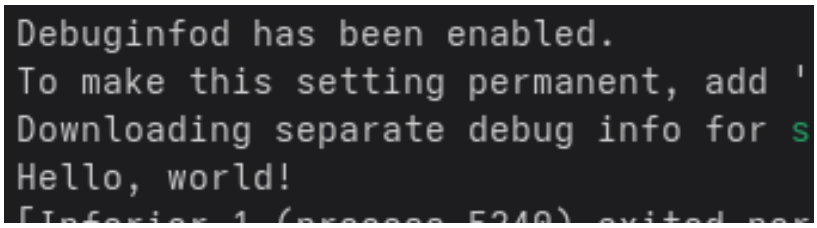

Рисунок 2.5: Скриншот терминала mc 5

Программа работает, как надо, перейдем к её анализу. Поставим брейкпоинт на старте программы.

Рисунок 2.6: Скриншот терминала mc 6

Посмотрим на диассимилированный код в двух синтаксисах.

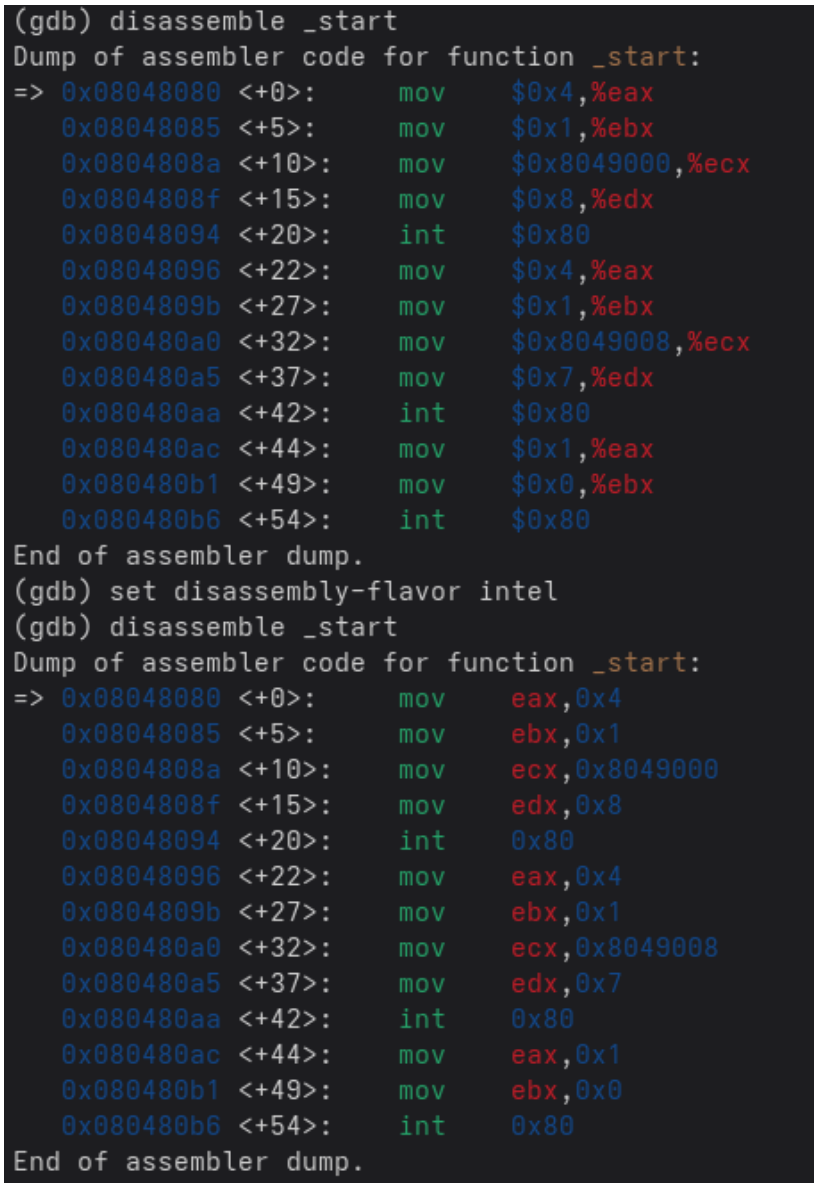

Рисунок 2.7: Скриншот терминала mc 7

Мы можем заметить разницу в отображении операндов инструкций в разных синтаксисах.

2.3 Добавление точек останова

Посмотрим на вид в псевдографике и список брейкпоинтов.

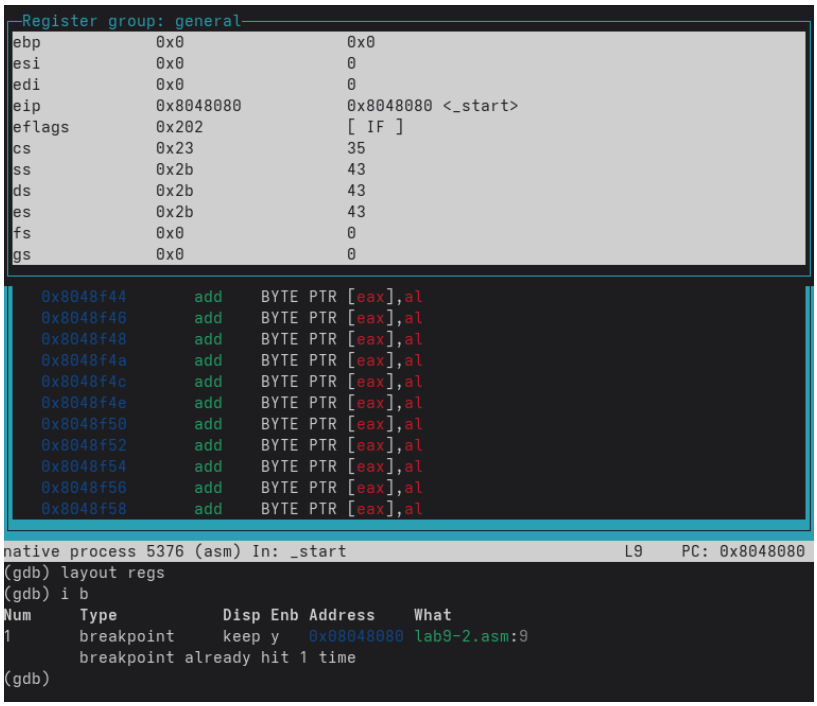

Рисунок 2.8: Скриншот терминала mc 8

Установим новый брейпоинт по адресу.

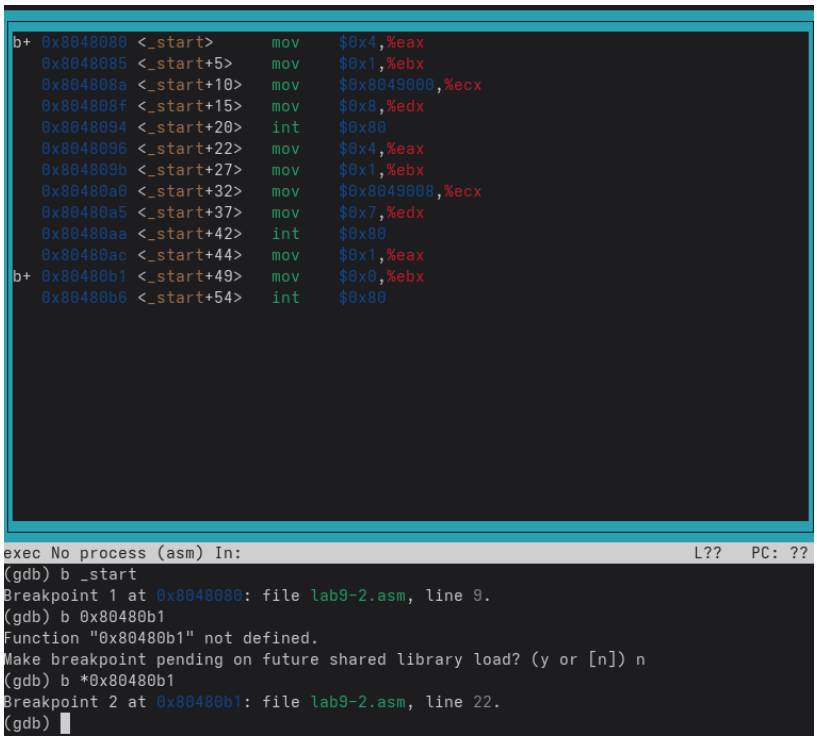

Рисунок 2.9: Скриншот терминала mc 9

2.4 Работа с данными программы в GDB

Посмотрим содержимое регистров

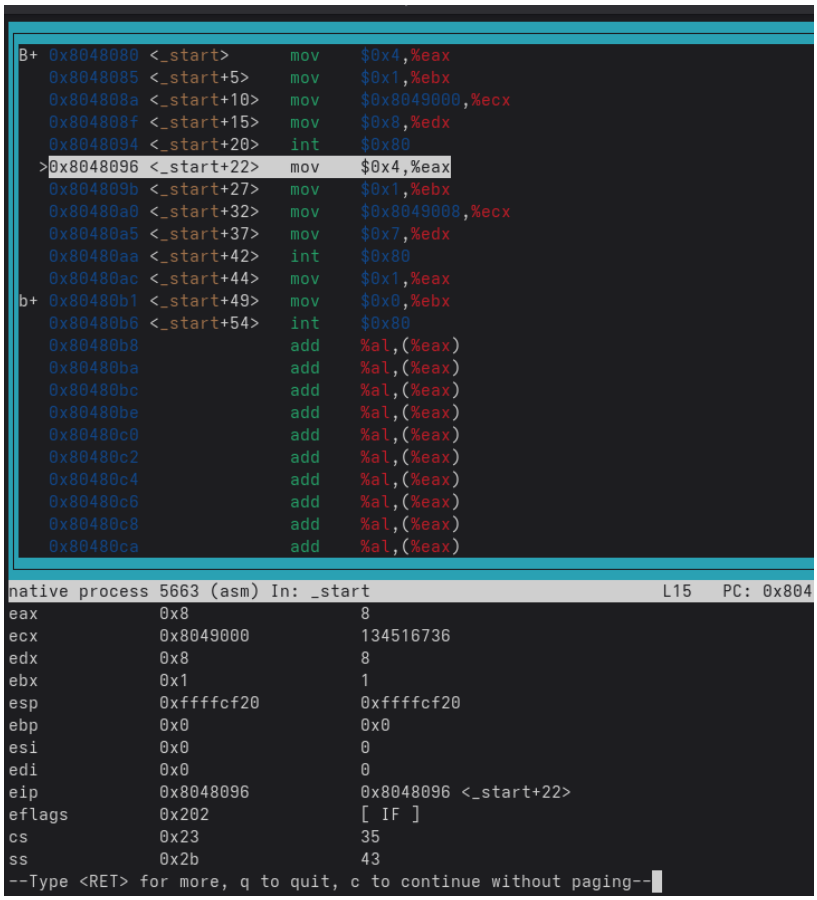

Рисунок 2.10: Скриншот терминала mc 10

Теперь посмотрим содержание переменных *msg1* и *msg2* по их адрессам.

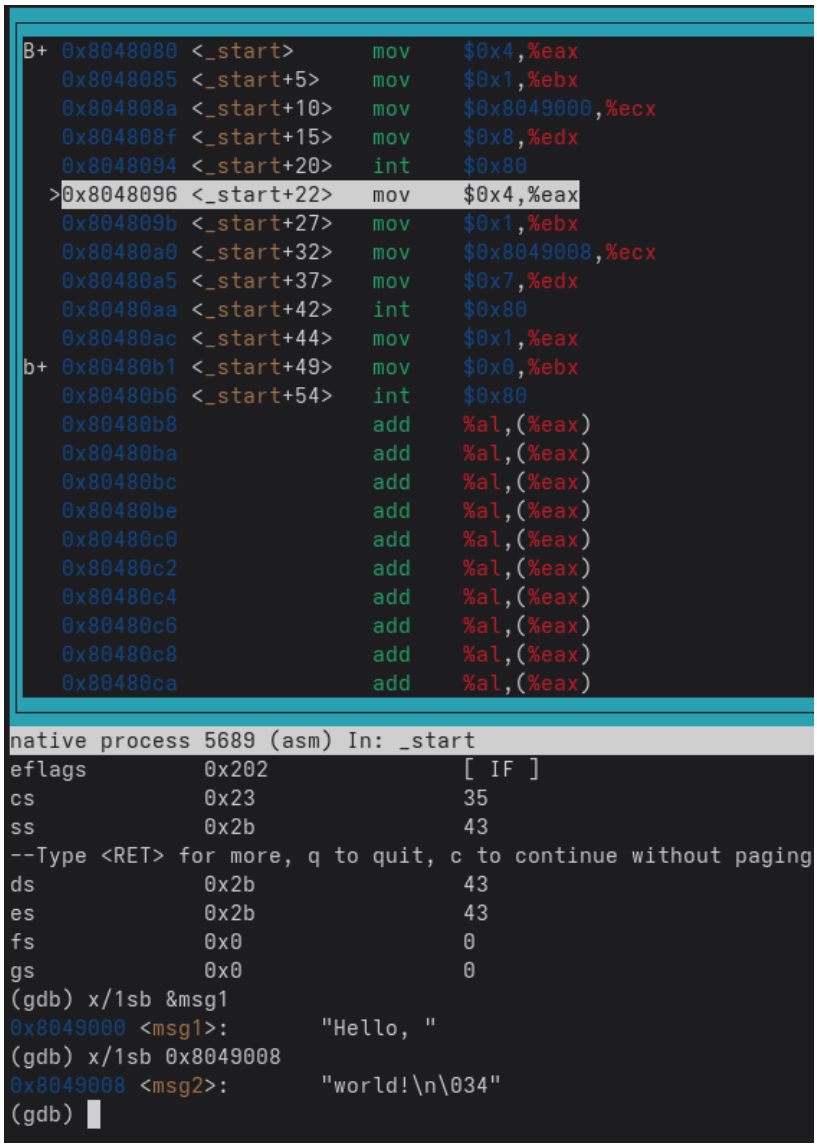

Рисунок 2.11: Скриншот терминала mc 11

П опробуем поменять символ в *msg2*. „w“ на „W“.

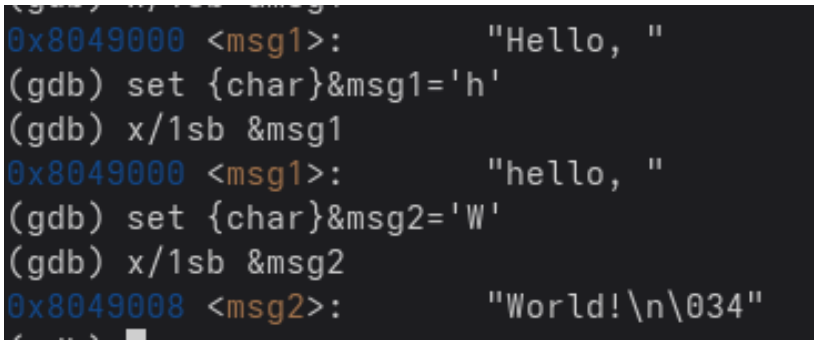

Рисунок 2.12: Скриншот терминала mc 12

Разница вывода команд *p/s $ebx* связана с тем, что сначала мы присвоили „2“, который имеет значение 50 в ASII коде, а потом число 2, которому соответсвует 2.

2.5 Обработка аргументов командной строки в GDB

Копируем файл *lab8-2.asm* с новым именем *lab9-3.asm*, компилируем и запускаем его с gdb.

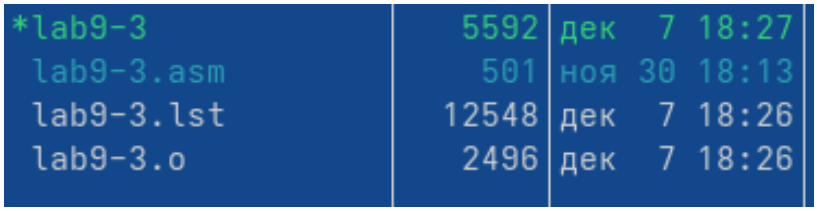

Рисунок 2.13: Скриншот терминала mc 13

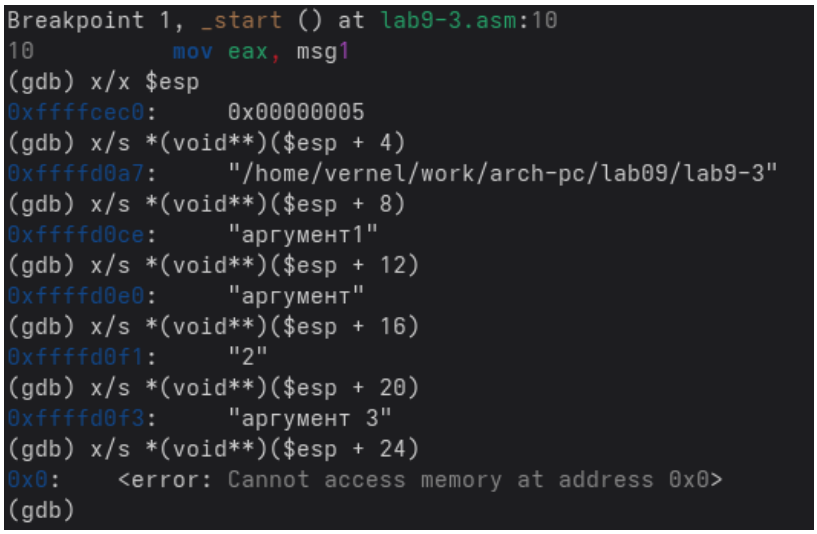

Рисунок 2.14: Скриншот терминала mc 14

Адрес аргументов увеличивается с шагом 4, так как каждый указатель занимает 4 байта.

3 Задание для самостоятельной работы

1. Копируем файл *self.asm* из прошлой лабораторной, записав вычисление функции в виде подпрограммы *_calc*, компилируем и запускаем.

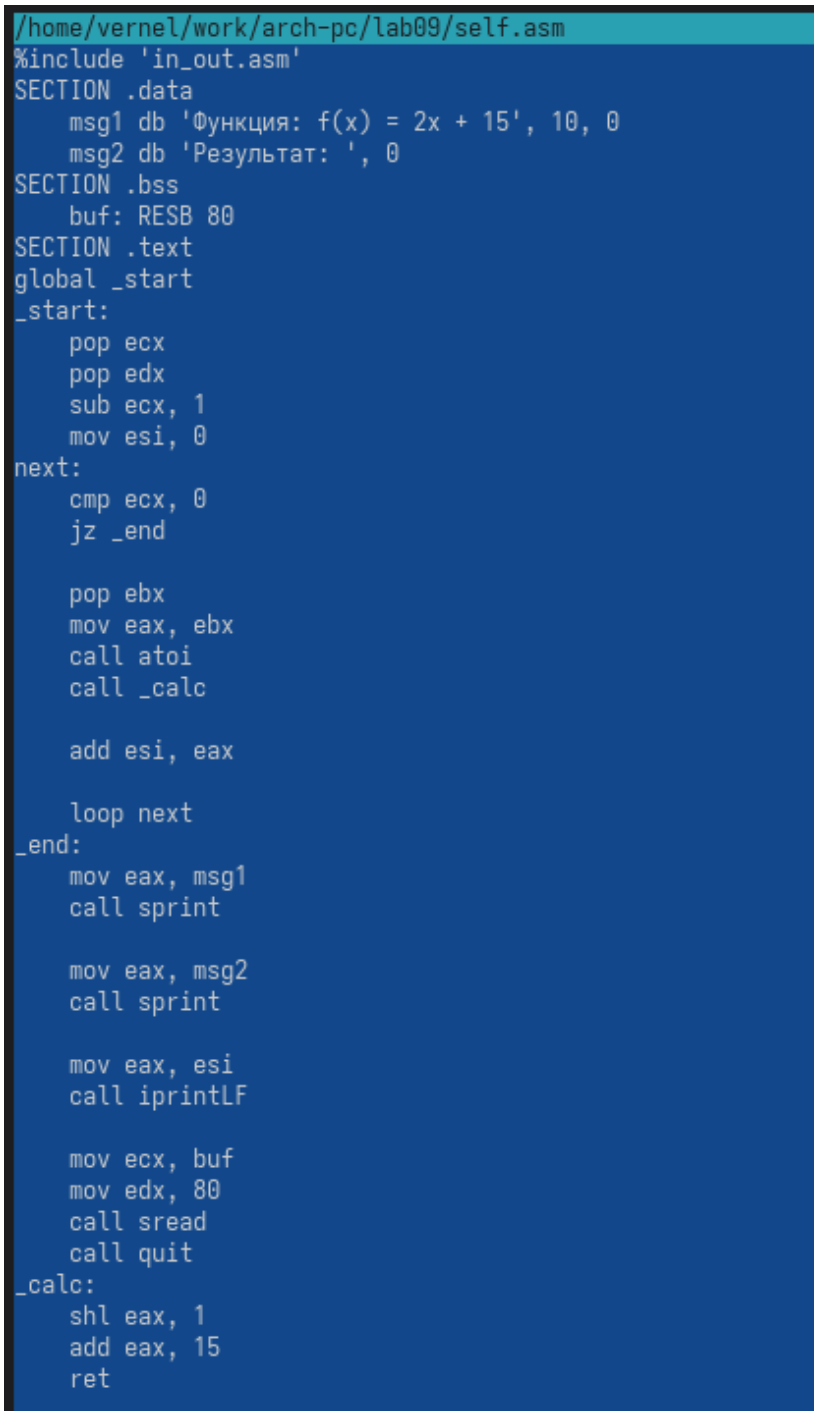

Рисунок 3.1: Скриншот терминала mc 15

```
zachnickif@fedora:~/work/arch-pc/lab09$ ./self 12
Функция: f(x) = 2x + 15
Результат: 39
```

Рисунок 3.2: Скриншот терминала mc 16

Программа работает корректно

2. В программе допущена ошибка, когда передаем в mul ecx, мы умножаем не ebx на 4, а ecx на 4, получаем 8, которая после не задействуется, поэтому выходит, что ebx+ebx=5+5=10, а не 25. Исправим программу.

```
self2.asm          [-M--] 17 L:[  1+14  15/
%include 'in_out.asm'
SECTION .data
div: DB 'Результат: ', 0

SECTION .text
GLOBAL _start
    _start:
<------>mov eax, 3
<------>add eax, 2
<------>
<------>mov ebx, 4
<------>mul ebx
<------>
<------>add eax, 5
<------>mov edi, eax
<------>
<------>mov eax, div
<------>call sprint
<------>
<------>mov eax, edi
<------>call iprintLF
<------>
<------>call quit
```

Рисунок 3.3: Скриншот терминала mc 17

Рисунок 3.4: Скриншот терминала mc 18

Теперь программа работает корректно.

4 Выводы

Были приобретены навыки написания программ с использованием подпрограмм. Познакомились с методами отладки при помощи GDB и его основными возможностями.