Отчёт по лабораторной работе №9

Кучерова Виктория Васильевна

Содержание

# 1 Цель работы

Приобрести навыки написания программ с использованием подпрограмм. Познакомиться с методами отладки при помощи GDB и его основными возможностями.

# 2 Выполнение лабораторной работы

Создадим каталог для выполнения лабораторной работы № 9, перейдем в него и создадим файл lab09-1.asm:(рис. 1).

Создание файла

Рис. 1: Создание файла

Введем в файл lab09-1.asm текст программы. Создадим исполняемый файл и проверим его работу(рис. 2).(рис. 3).

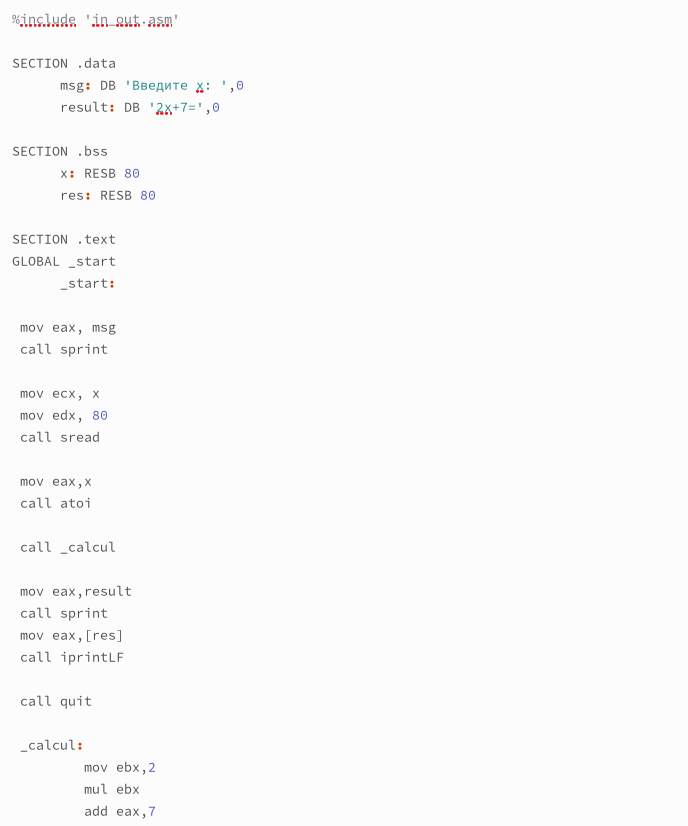


Рис. 2: Программа

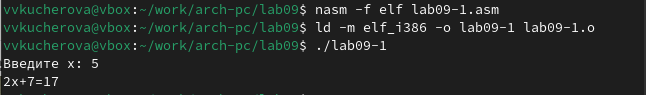


Рис. 3: Запуск

Изменим текст программы, добавив подпрограмму \_subcalcul в подпрограмму \_calcul, для вычисления выражения f(g(x)), где x вводится с клавиатуры(рис. 4).(рис. 5).

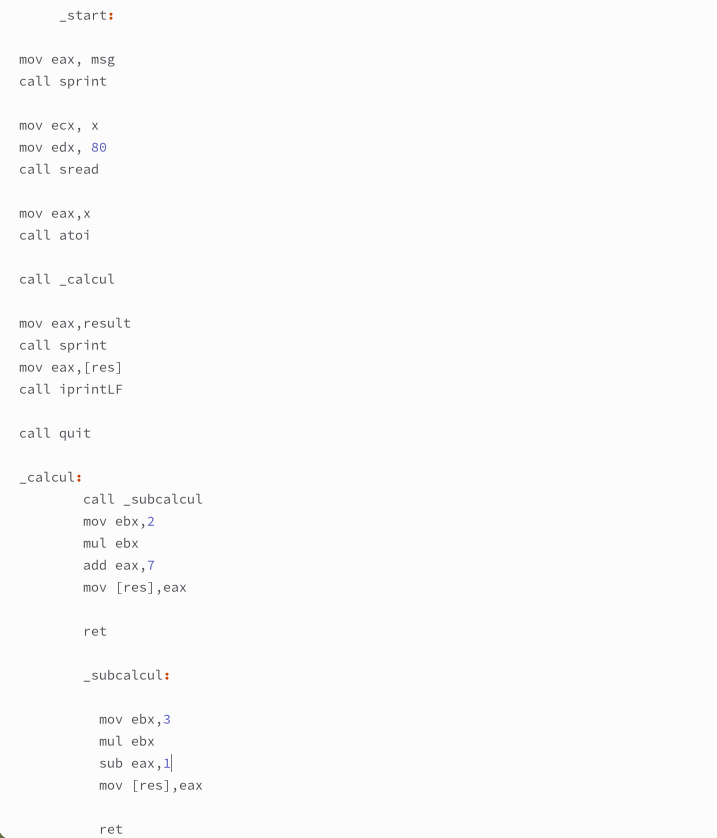


Рис. 4: Программа

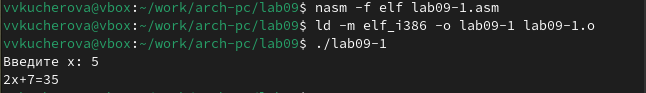


Рис. 5: Запуск

Создадим файл lab09-2.asm с текстом программы(рис. 6).

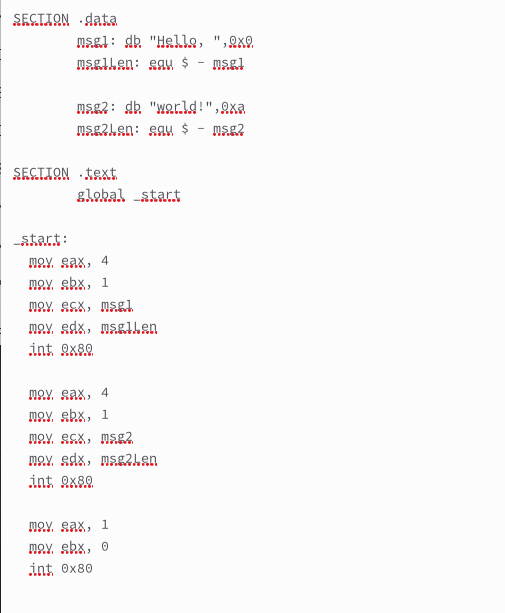


Рис. 6: Программа

Получим исполняемый файл и загрузим исполняемый файл в отладчик gdb(рис. 7).

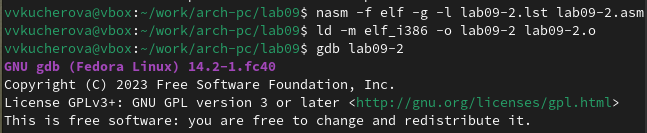


Рис. 7: Исполняемый файл

Проверим работу программы, запустив ее в оболочке GDB с помощью команды run(рис. 8).

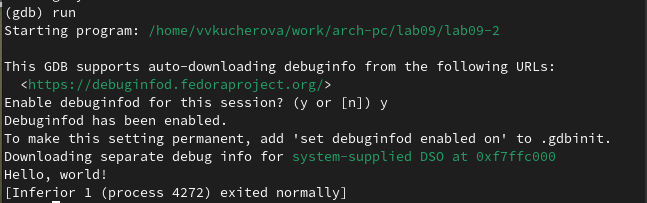


Рис. 8: Команда run

Для более подробного анализа программы установим брейкпоинт на метку \_start(рис. 9).

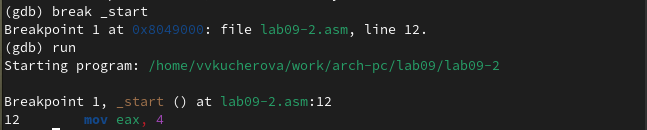


Рис. 9: брейкпоинт

Посмотрим дисассимилированный код программы с помощью команды disassemble начиная с метки \_start(рис. 10).

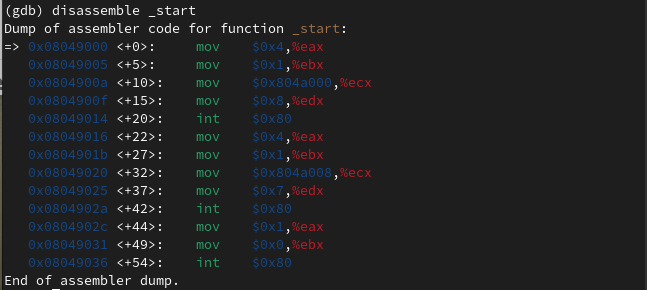


Рис. 10: Дисассимилированный код

Переключитесь на отображение команд с Intel’овским синтаксисом. В режиме ATT регистры указываются после их адреса и перед регистром стоит %(рис. 11).

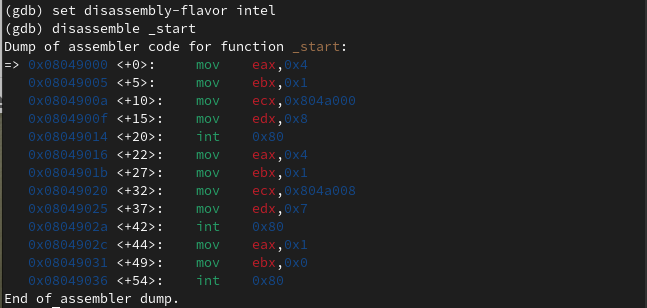


Рис. 11: Intel’овский синтаксис

Включим режим псевдографики для более удобного анализа программы(рис. 12).

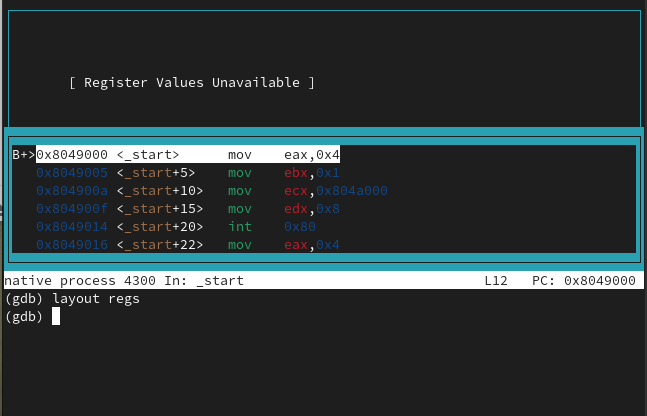


Рис. 12: Режим псевдографики

Ранее была установлена точка останова по имени метки (\_start). Проверим это с помощью команды info breakpoints(рис. 13).

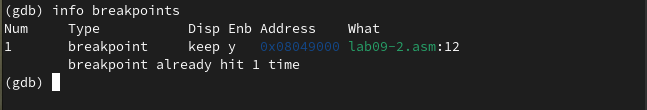


Рис. 13: Команда info breakpoints

Установим еще одну точку останова по адресу инструкции(рис. 14).

Еще одна точка останова

Рис. 14: Еще одна точка останова

Посмотрим информацию о всех установленных точках останова(рис. 15).

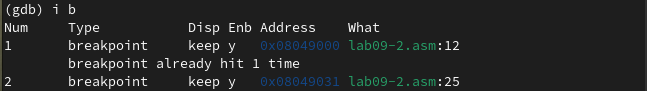


Рис. 15: Точки останова

Выполним 5 инструкций с помощью команды stepi(рис. 16).(рис. 17).(рис. 18).(рис. 19).(рис. 20).

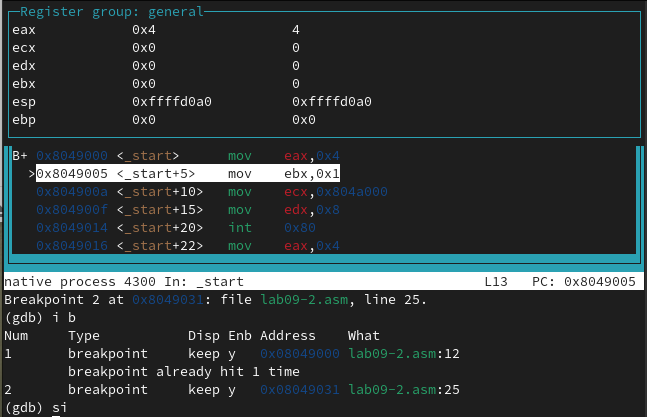


Рис. 16: stepi 1

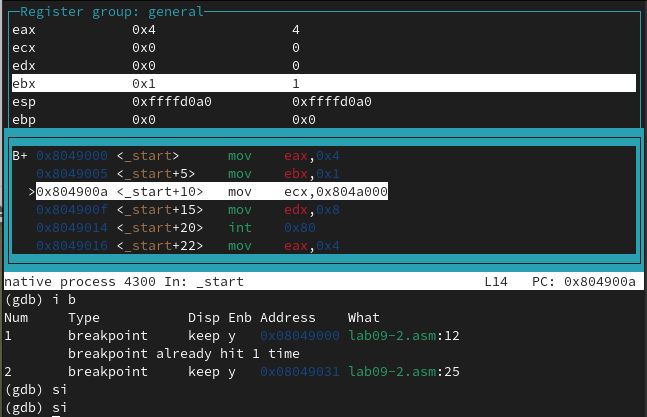


Рис. 17: stepi 2

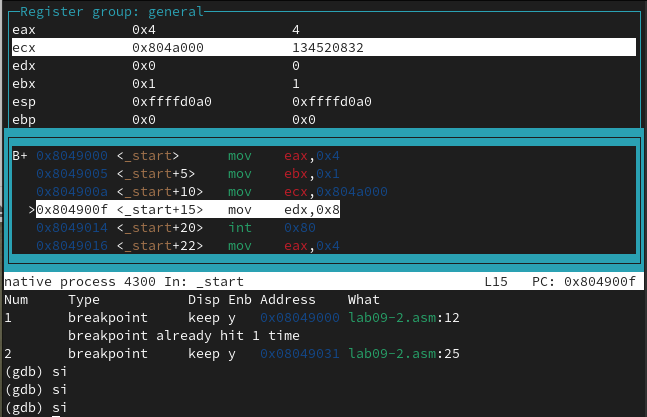


Рис. 18: stepi 3

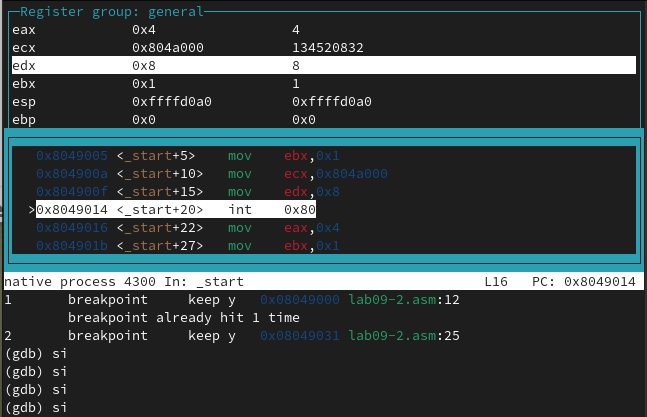


Рис. 19: stepi 4

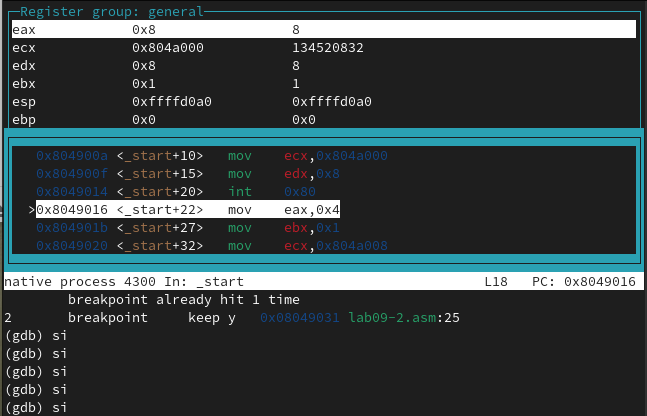


Рис. 20: stepi 5

Посмотрим значение переменной msg1 по имени(рис. 21).

Значение переменной msg1

Рис. 21: Значение переменной msg1

Посмотрим значение переменной msg2 по адресу(рис. 22).

Значение переменной msg2

Рис. 22: Значение переменной msg2

Изменим первый символ переменной msg1(рис. 23).

Изменение msg1

Рис. 23: Изменение msg1

Заменим любой символ во второй переменной msg2(рис. 24).

Изменение msg2

Рис. 24: Изменение msg2

Выведем в различных форматах значение регистра edx(рис. 25).



Рис. 25: Значение регистра edx

С помощью команды set изменим значение регистра ebx. В первом случае мы вносим значение 2, а во втором регистр равен 2(рис. 26).



Рис. 26: Изменение значения регистра ebx

Выйдем из GDB(рис. 27).



Рис. 27: Выход

Скопируем файл lab8-2.asm в файл с именем lab09-3.asm и создадим исполняемый файл(рис. 28).

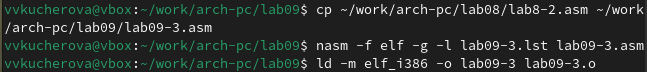


Рис. 28: Копирование

Загрузим исполняемый файл в отладчик, указав аргументы(рис. 29).

Отладчик

Рис. 29: Отладчик

Для начала установим точку останова перед первой инструкцией в программе и запустим ее(рис. 30).

Запуск

Рис. 30: Запуск

Адрес вершины стека храниться в регистре esp и по этому адресу располагается число равное количеству аргументов командной строки(рис. 31).

Регистр esp

Рис. 31: Регистр esp

Посмотрим остальные позиции стека. Элементы расположены с интервалом 4, так как стек может хранить до 4 байт, и для того чтобы данные сохранялись нормально без помех, компьютер использует новый стек для новой информации(рис. 32).

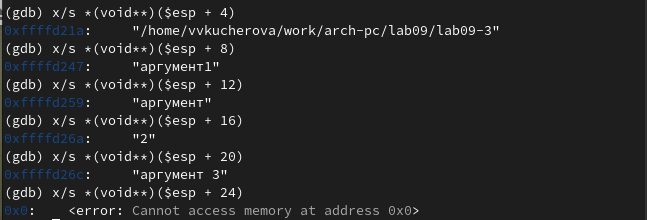


Рис. 32: Остальные позиции стека

# 3 Задание для самостоятельной работы

Преобразуем программу из лабораторной работы №8, реализовав вычисление значения функции f(x) как подпрограмму(рис. 33).(рис. 34).

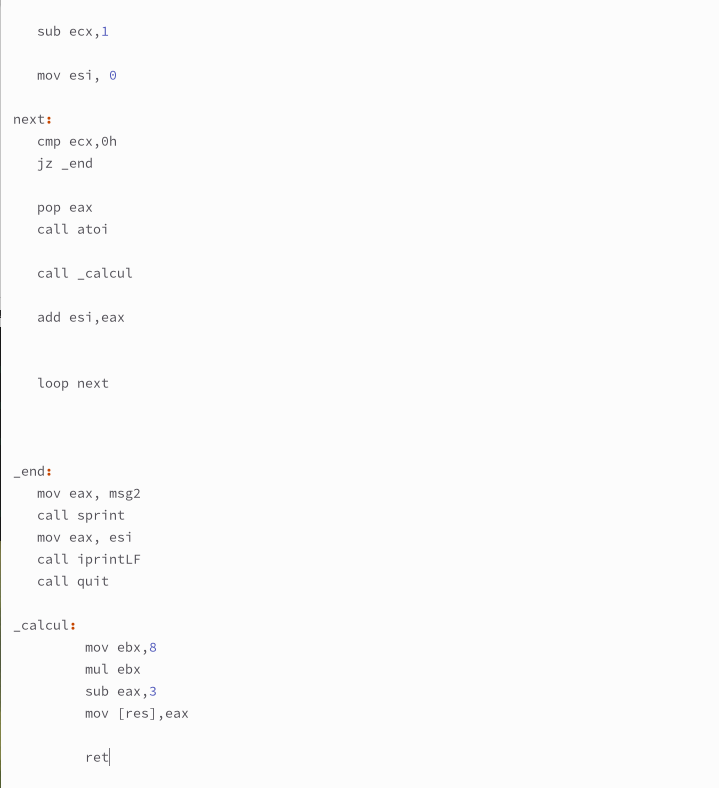


Рис. 33: Программа

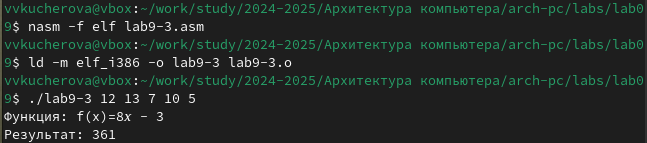


Рис. 34: Запуск

При запуске данная программа дает неверный результат. Проверим это. С помощью отладчика GDB, анализируя изменения значений регистров, определим ошибку и исправьте ее. Ошибка в том что сумма записывается в ebx, а на 4 умножается eax(рис. 35).(рис. 36).(рис. 37).(рис. 38).

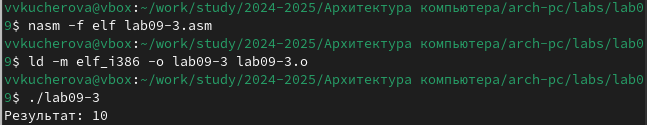


Рис. 35: Неверный результат

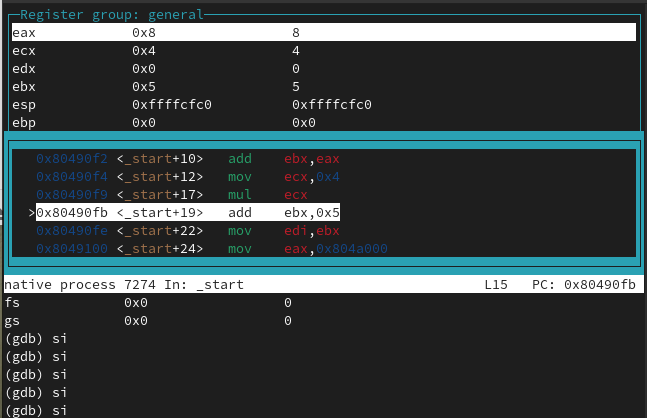


Рис. 36: Ошибка

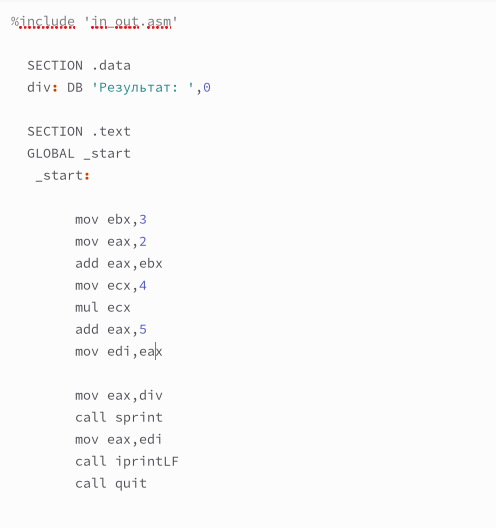


Рис. 37: Исправление

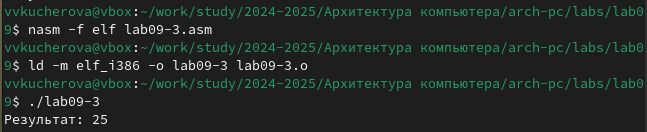


Рис. 38: Запуск

# 4 Выводы

Я приобрела навыки написания программ с использованием подпрограмм. Познакомилась с методами отладки при помощи GDB и его основными возможностями.

# Список литературы