Лабораторная работа №9

Дисциплина: Архитектура компьютера

Жибицкая Евгения Дмитриевна

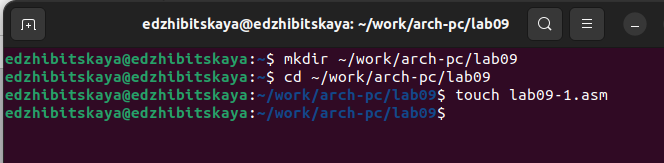
Содержание

# 1 Цель работы

Продолжение изучения языка ассемблера, использование подпрограмм и знакомство с методами отладки при помощи GDB.

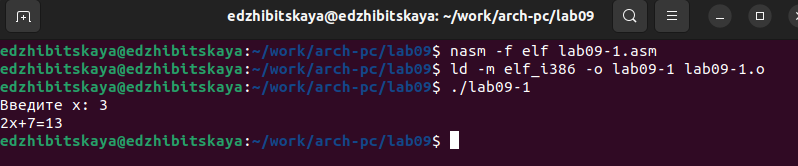
# 2 Выполнение лабораторной работы

Для начала создадим каталог для 9 лабораторной работы, создадим файл и не забудем скопировать файл in\_out.asm в этот же каталог(рис. ??).



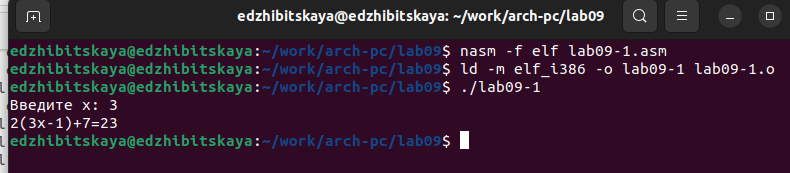
Создание каталога и файла

Затем заполним файл текстом из Листинга 9.1, создадим исполняемый файл и запустим(рис. ??).



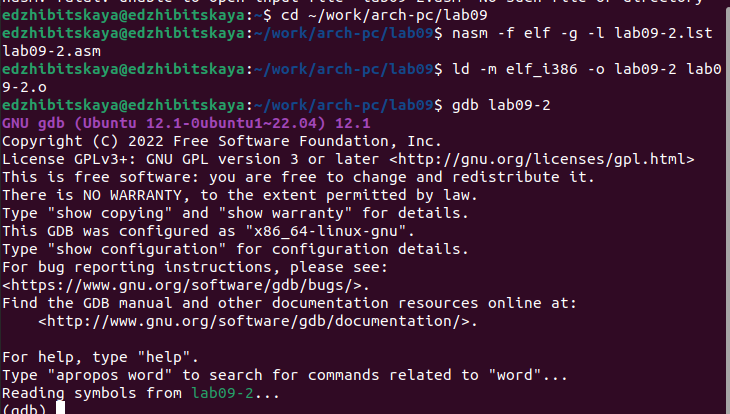
Запуск программы

Потом изменим программу, добавив подпрограмму \_subcalcul и также запустим ее(рис. ??).



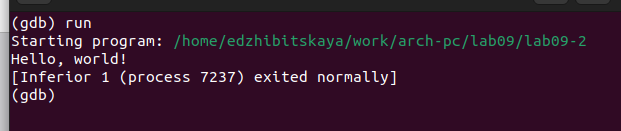
Создание подпрограммы

Далее добавим файл lab09-2.asm с текстом программы из Листинга 9.2, создадим исполняемый файл с ключом -g для отладочной информации и загрузим его в отладчик(рис. ??).

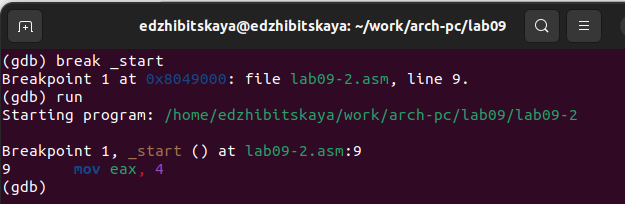


Загрузка в отладчик(и создание файла)

Затем командой run запустим программу(рис. ??), также добавим брейкпоинт и еще раз запустим ее(рис. ??).

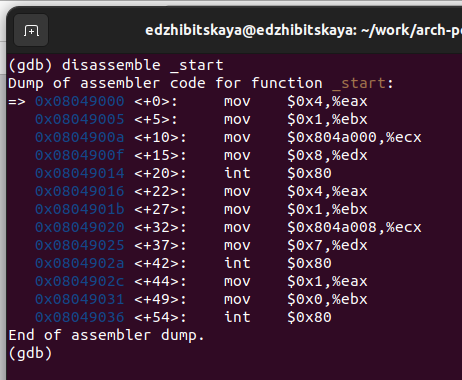


Запуск программы

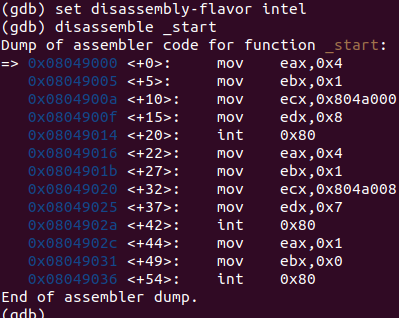


Установка брейкпоинта и запуск

Посмотрим дисассимилированный код программы командой disassemble(рис. ??), также переключимся на отображение с Intel’овским синтаксисом(рис. ??). Отличаются порядком операндов, записью числовых констант, наличием/отсутствием % перед именем регистра и тд.

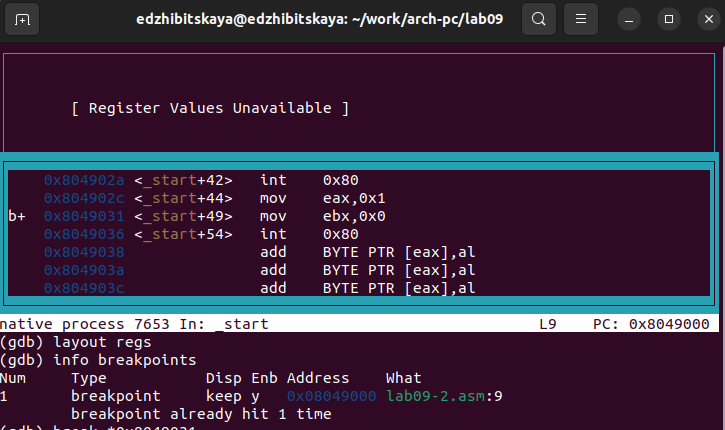


Дисассимилированный код



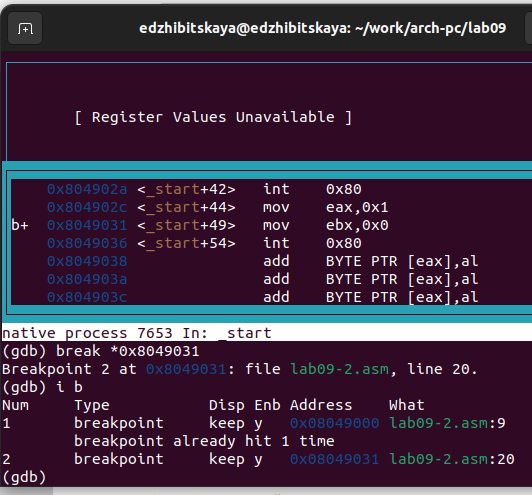
Дисассимилированный код с Intel’овским синтаксисом

Далее включим режим псевдографики и сразу посмотрим уже установленные ранее точки останова(рис. ??)



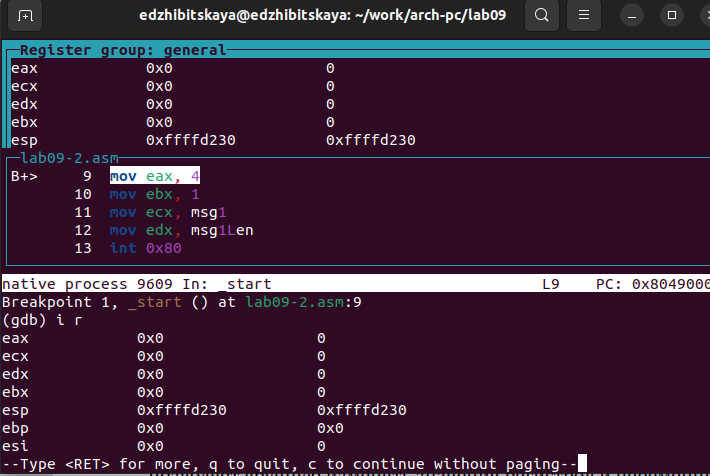
Режим псевдографики

Установим еще один брейкпоинт и еще раз проверим информацию о всех точках останова(рис. ??).



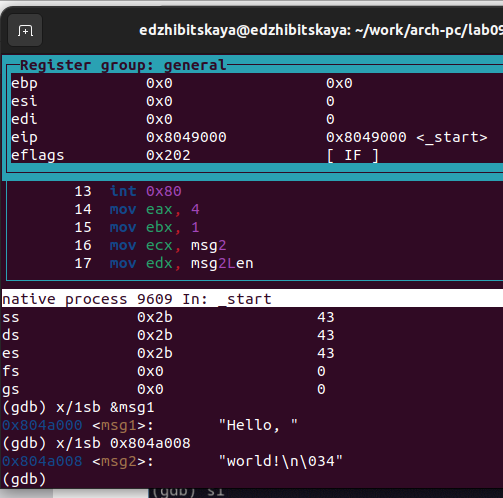
Установка и просмотр точек останова

Чтобы посмотреть содержимое регистров введем info registers(рис. ??).



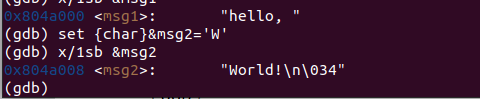
info registers

Затем посмотрим значение переменной msg1 по имени и msg2 по адресу(рис. ??).



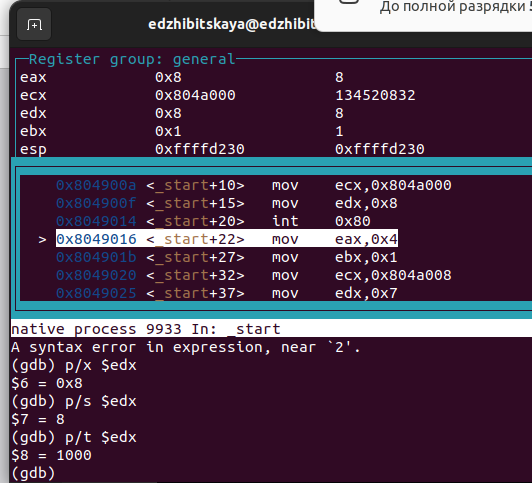
Просмотр значений переменных

После инструкцией set изменим символы в переменных msg1 и msg2(рис. ??)

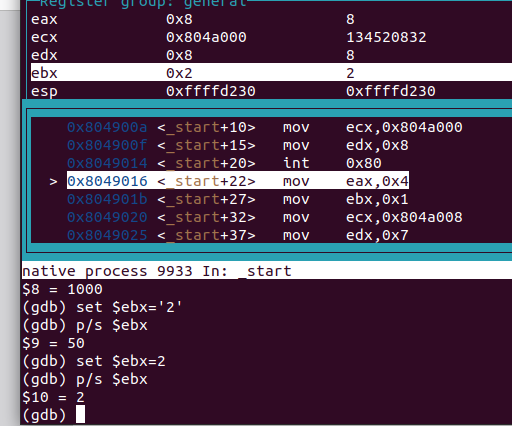


Изменение символов в переменных

Наконец командой print посмотрим значения регистра(в 16ричном,2оичном и символьном форматах)(рис. ??) и изменим значение регистра ebx(симол и число)(рис. ??).

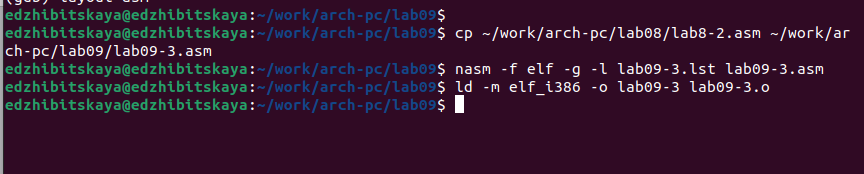


Просмотр значения edx



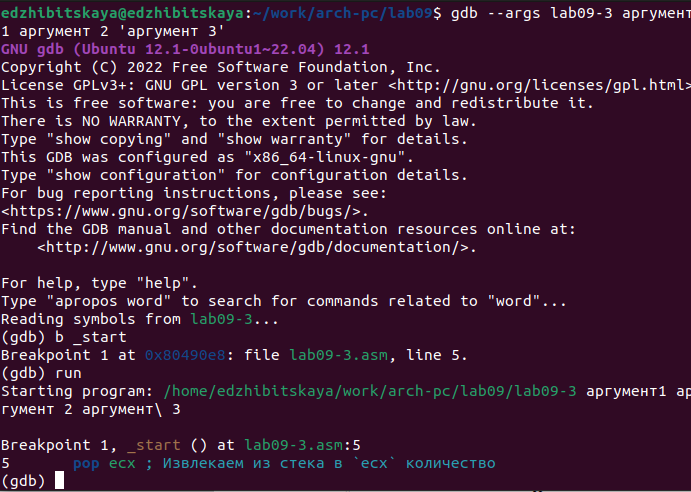
Изменение регистра еbx

Следующим этапом скопируем файл из предыдущией работы и создадим исполняемый файл(рис. ??).



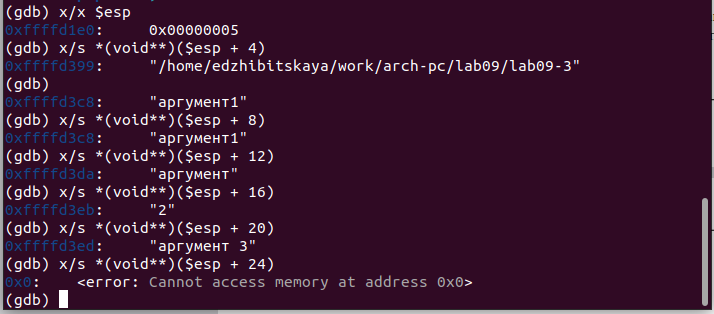
Копирование файла

Загрузим эту программу в отладчик использовав ключ –args, установим точку останова и запустим ее(рис. ??).



Запуск прораммы в отладчике

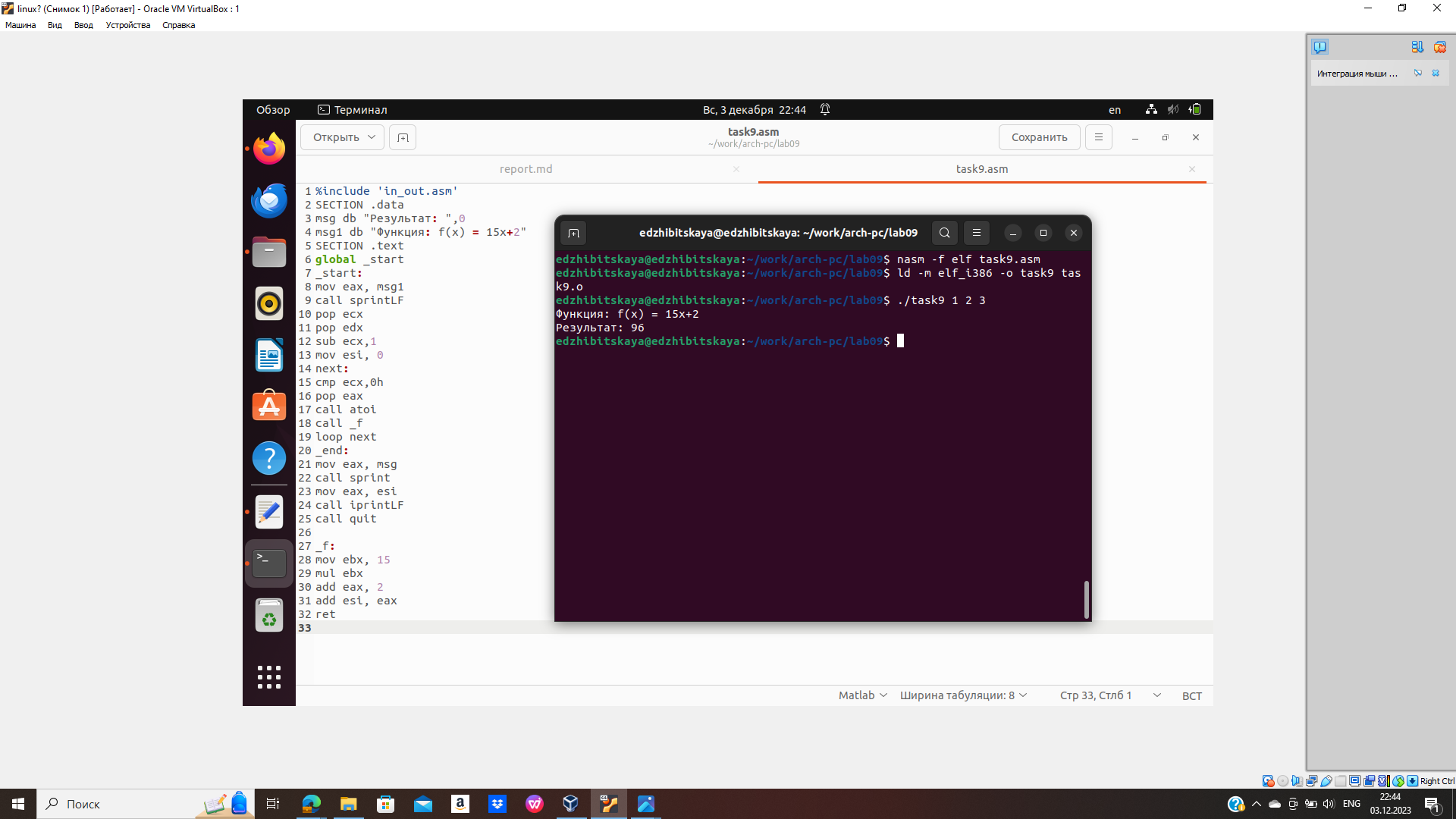
В конце посмотрим, что число аргументов равно 5 - посмотрим остальные позиции стека по адресу(рис. ??).



Просмотр позиций стека

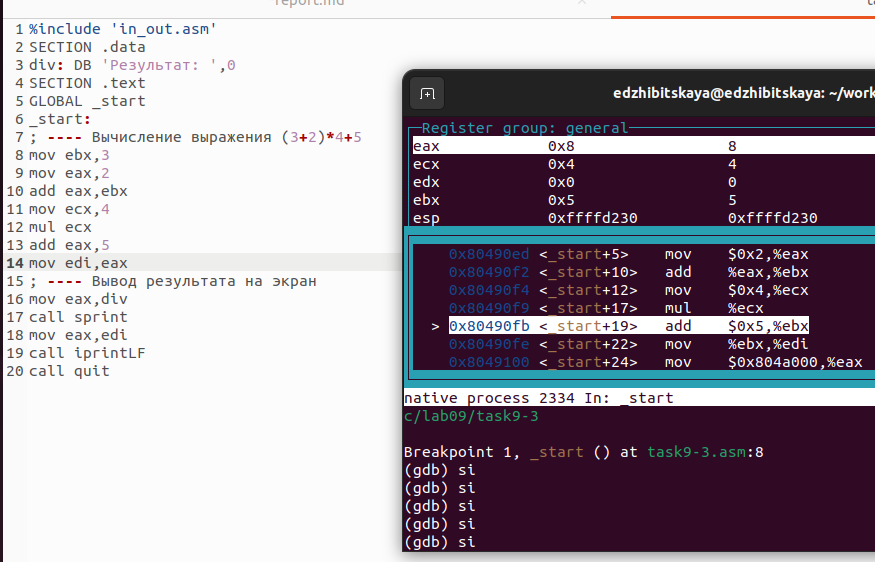
# 3 Задание для самостоятельной работы

1. Преобразуем программу из 8 лабораторной работы, реализовав вычисление функции как подпрограммы(рис. ??).



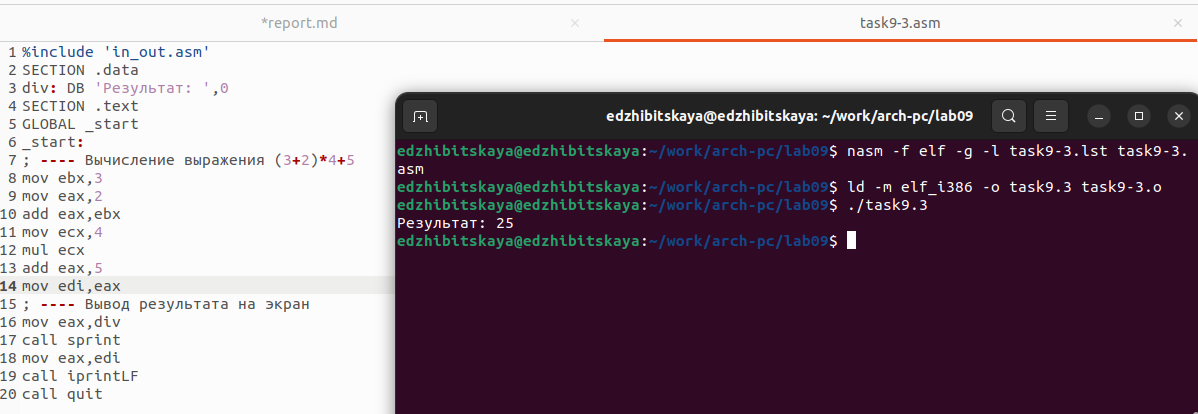
Код и запуск программы

1. Определение и исправление ошибки из Лиситнга 9.3 C помощью отладчика gdb, поочередно выполняя команды и следя за значениями регистров, видим, что ошибка возникает из-за неправильной записи резултата сложения в регистр ebx, а не eax, так как при последующем умножении, умножается именно регистр eax(рис. ??).



Поиск ошибки

Исправим код программы и запустим ее(рис. ??).



Код и запуск программы

# 4 Выводы

В ходе работы было произведено знакомство с методами и возможностями отладки программ, применены подпрограммы.