Лабораторная работа №14

Средства, применяемые приразработке программного обеспечения в ОС типа UNIX/Linux.

Акопян Изабелла Арменовна

Содержание

# Цель работы

Приобрести простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.

# Задание

1. Создать каталог ~/work/os/lab\_prog.
2. Создать в нём файлы: calculate.h, calculate.c, main.c.
3. Выполнить компиляцию.
4. Создать makefile.
5. Выполнить отладку программы calcul.
6. Проанализировать коды файлов calculate.c и main.c.

# Выполнение лабораторной работы

Ознакомилась с лабораторной работой №13 ([ссылка 1](https://esystem.rudn.ru/pluginfile.php/1142526/mod_resource/content/2/011-lab_prog.pdf)) и приступила к выполнению заданий.

В домашнем каталоге создала подкаталог ~/work/os/lab\_prog.

В каталоге создала файлы: calculate.h, calculate.c, main.c. Это нужно для примитивнейшего калькулятора, способного складывать, вычитать,умножать и делить, возводить число в степень, брать квадратный корень, вы-числять sin, cos, tan. При запуске он будет запрашивать первое число, операцию, второе число. После этого программа должна будет вывести результат и остановится.

Ниже показаны команды запуска редактора emacs (рис. 1).Там же и на (рис. 2) показана компиляция программ с помощью gcc (опция -c создает объектные файлы, -o задает имена файлам).

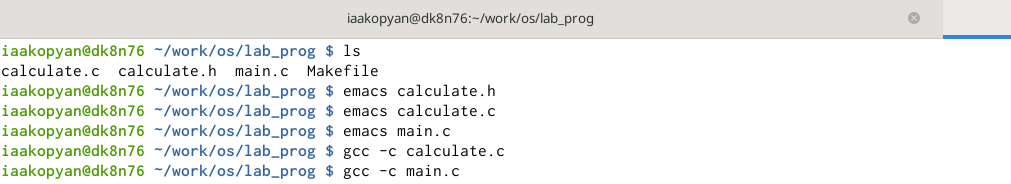


Figure 1: Команды

Figure 2: Команды

Figure 2: Команды

В файл main.c записала этот код (рис. 3):

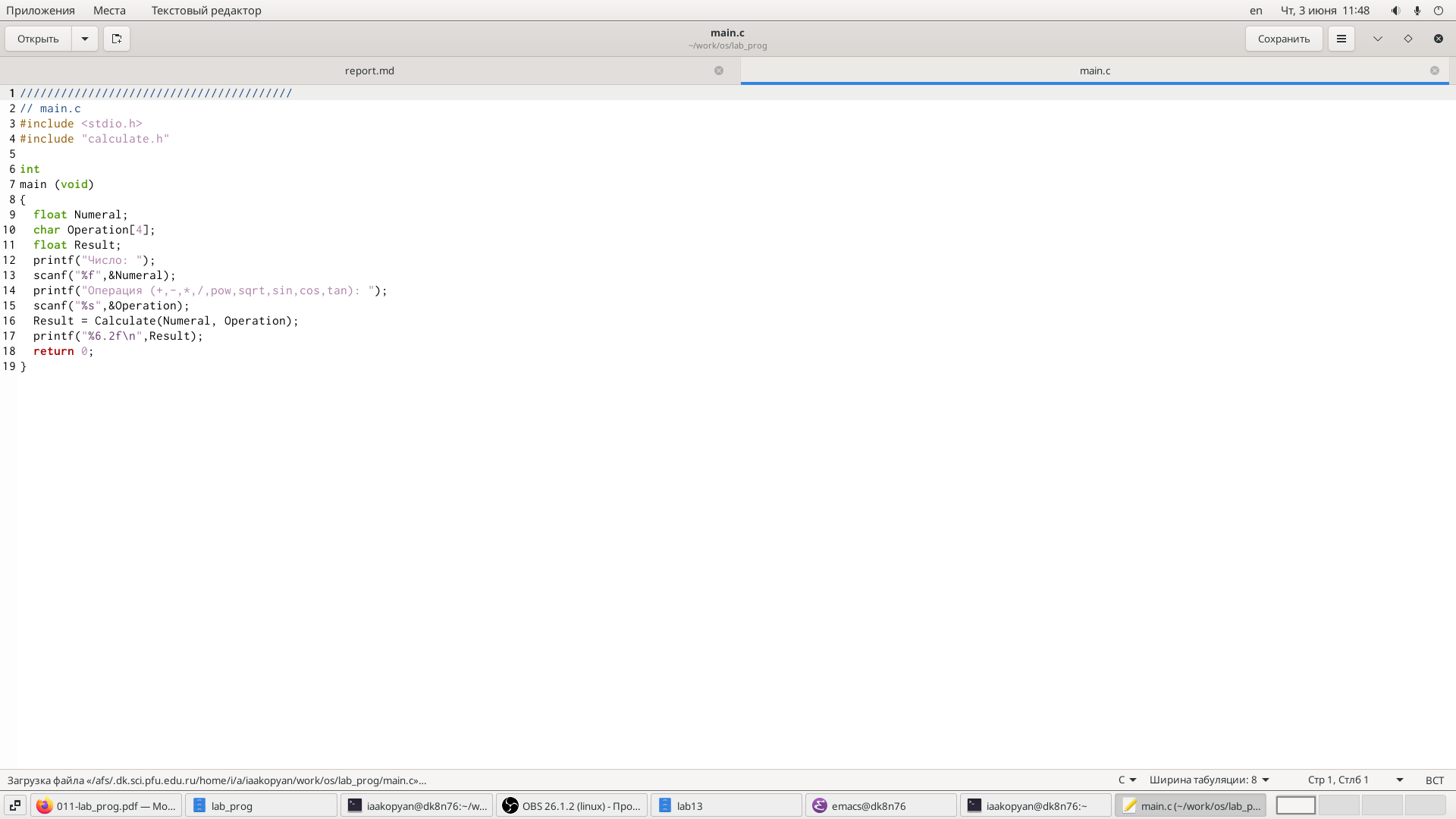


Figure 3: main.c

В файл calculate.c записала этот (рис. 4):

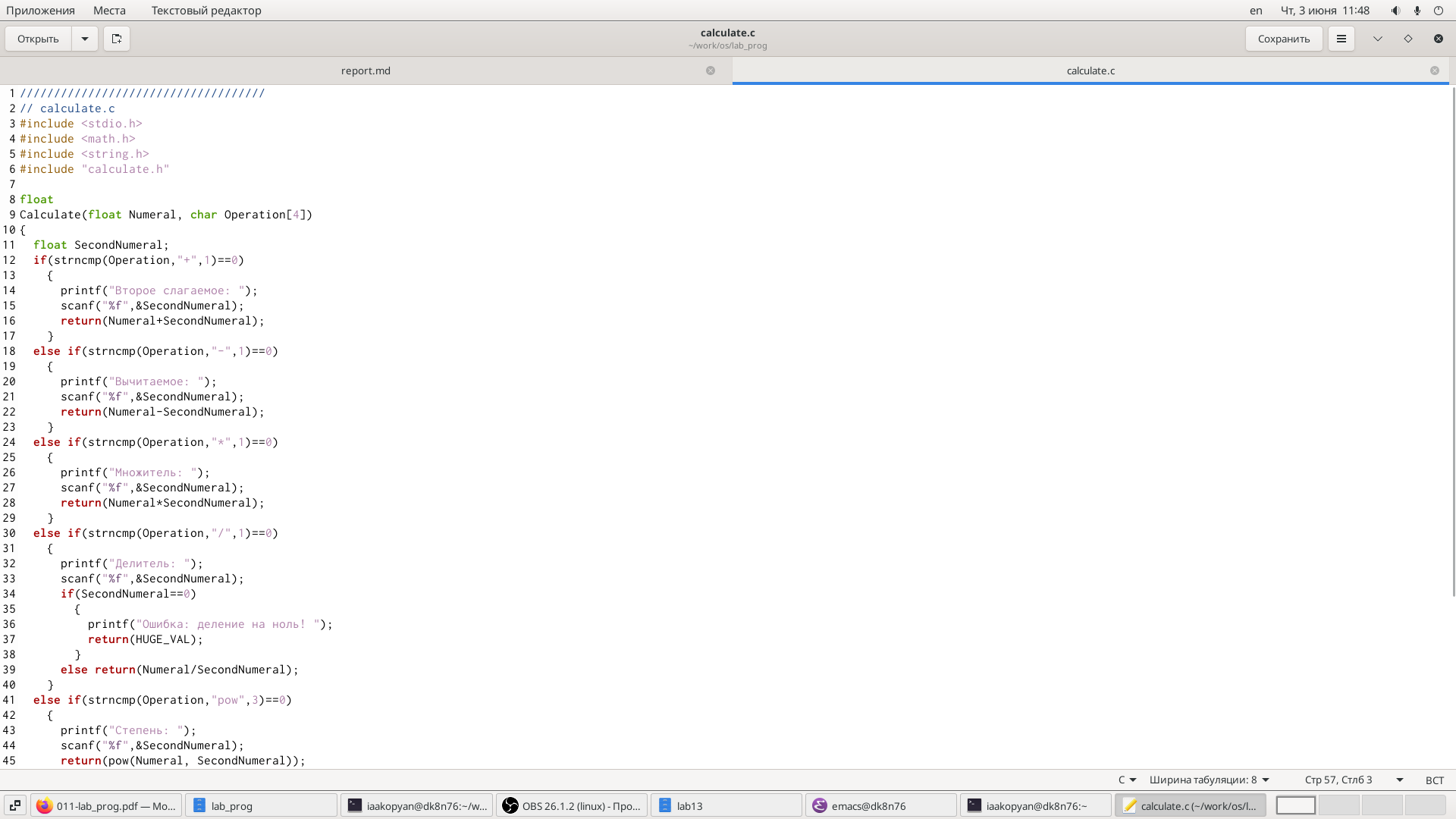


Figure 4: calculate.c

Содержимое файла calculate.h (рис. 5):

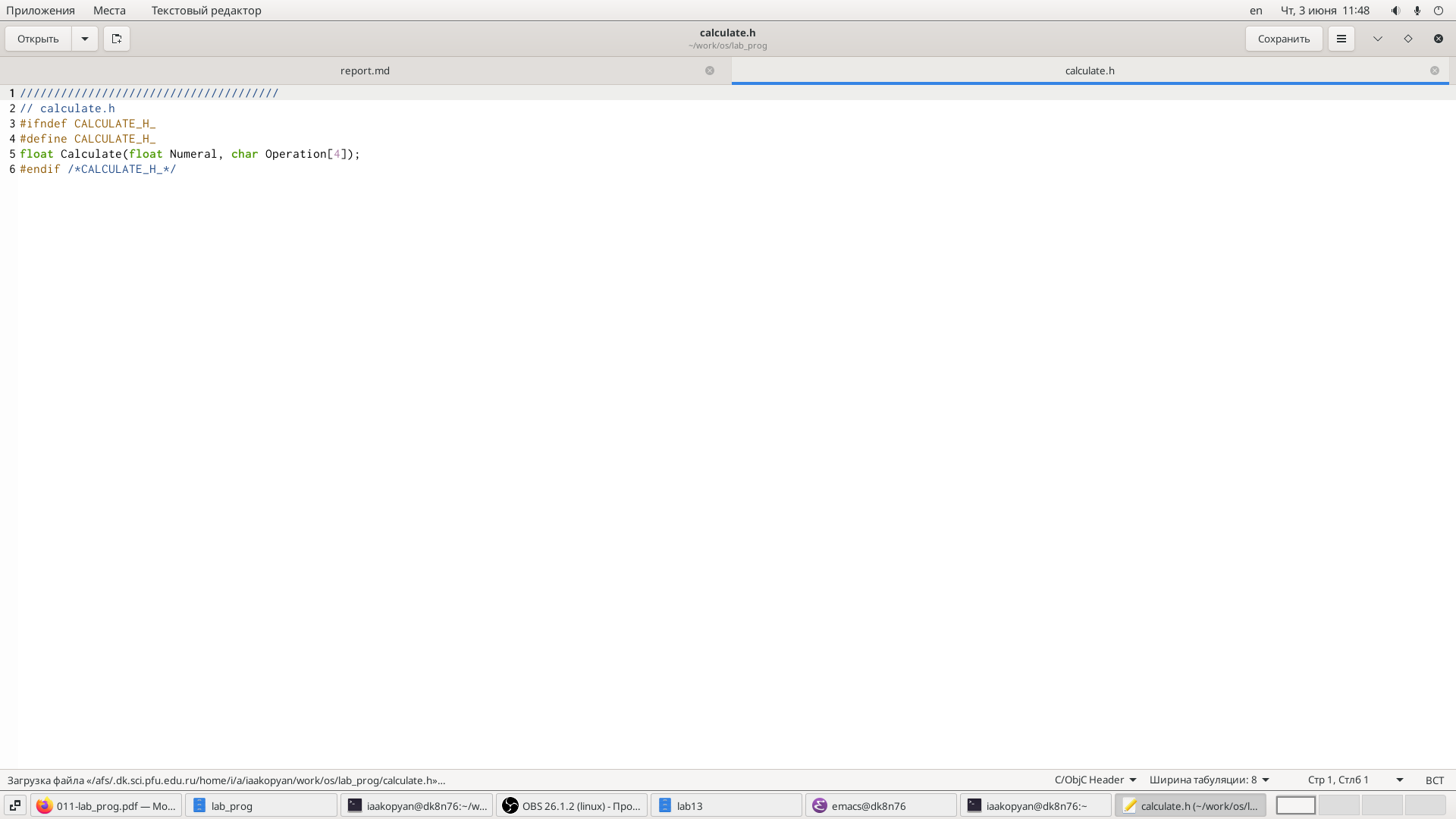


Figure 5: calculate.h

Создала Makefile (рис. 6):

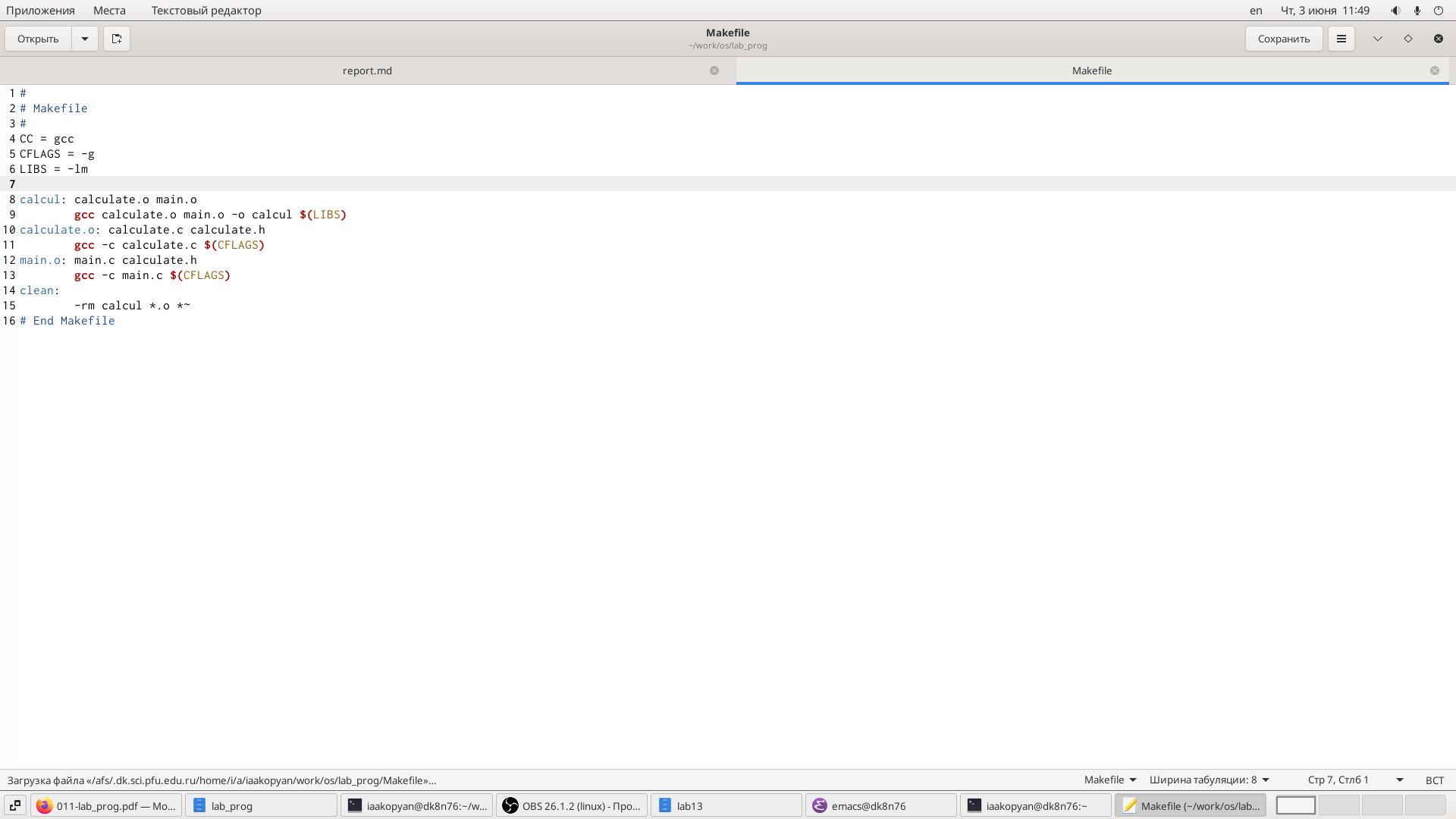


Figure 6: makefile

С помощью gdb выполнила отладку программы calcul (рис. 7). Чтобы запустить отладчик пишу ‘gdb ./calcul’. Для запуска калькулятора внутри отладчика пишу ‘run’.

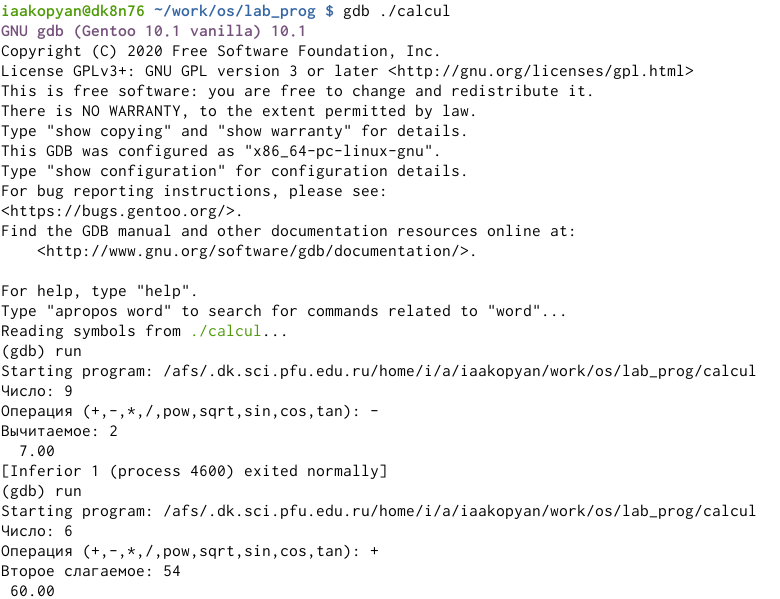


Figure 7: GDB

Чтобы просмотреть первые 10 строк кода пишу ‘list’. Или же использую лист с параметрами. Для точки останова пишу ‘break ’. (рис. 8)



Figure 8: GDB

Запустила программу внутри отладчика, программа остановилась в момент прохождения точки останова (рис. 9). Отладчик выдал следующую информацию:

‘#0 Calculate (Numeral=5, Operation=0x7fffffffcf14 “-”)at calculate.c:21’

‘#1 0x0000000000400c31 in main () at main.c:16’

Я вызвала команду backtrace и смогла вывести весь стек вызываемых функций от начала программы до текущего места.

Посмотрела, чему равно на этом этапе значение переменной Numeral, введя:

print Numeral

Было выведено число 5.

Сравнила с результатом вывода на экран после использования команды:

display Numeral

Удалила точку останова (delete 1).

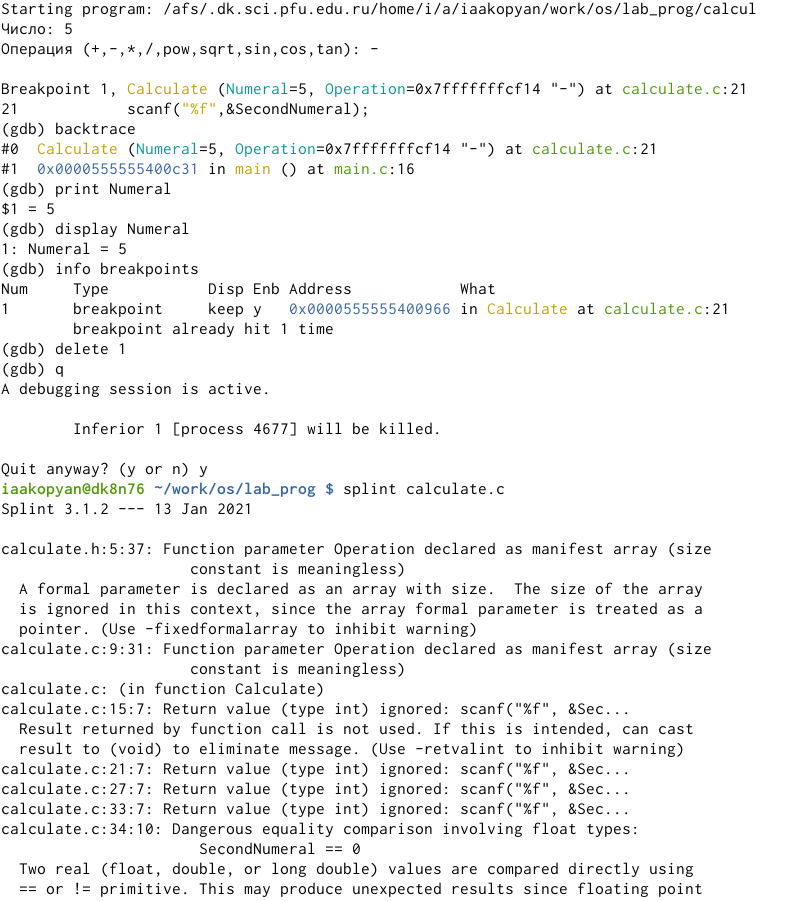


Figure 9: GDB и splint

С помощью утилиты splint попробовала проанализировать коды файлов calculate.c и main.c (рис. 9) (рис. 10). Информацию о splint нашла здесь: [ссылка 2](http://espressocode.top/splint-c-program-verifier/)

Splint- это UNIX программа, позволяющая проводить статический анализ кода, написанного на Си.

Splint особенно хорош при проверке проверки типов назначений переменных и функций, эффективности, неиспользуемых переменных и идентификаторов функций, недоступного кода и возможных утечек памяти.

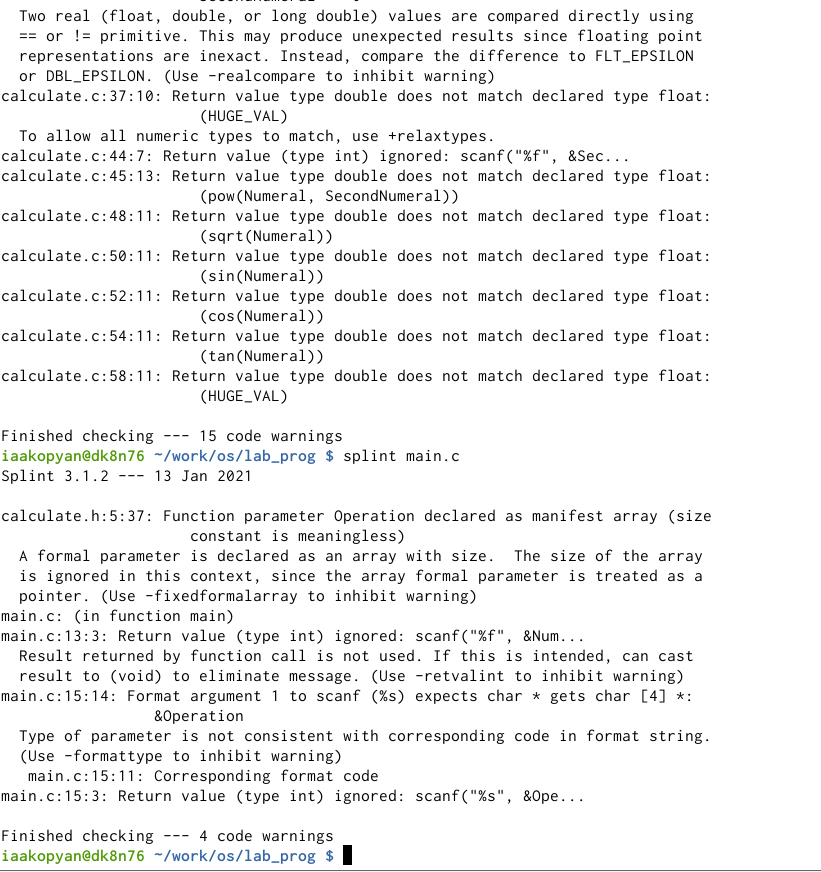


Figure 10: splint

# Выводы

Приобрела простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.

# Библиография

[ссылка 1](https://esystem.rudn.ru/pluginfile.php/1142526/mod_resource/content/2/011-lab_prog.pdf)

[ссылка 2](http://espressocode.top/splint-c-program-verifier/)