Лабораторная работа №13

Дисциплина - операционные системы

Волгин Иван Алексеевич

Содержание

# 1 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.

# 2 Задание

1. В домашнем каталоге создайте подкаталог ~/work/os/lab\_prog.
2. Создайте в нём файлы: calculate.h, calculate.c, main.c. Это будет примитивнейший калькулятор, способный складывать, вычитать, умножать и делить, возводить число в степень, брать квадратный корень, вычислять sin, cos, tan. При запуске он будет запрашивать первое число, операцию, второе число. После этого программа выведет результат и остановится.
3. Выполните компиляцию программы посредством gcc:
4. При необходимости исправьте синтаксические ошибки.
5. Создайте Makefile.
6. С помощью gdb выполните отладку программы calcul (перед использованием gdb исправьте Makefile):
7. С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c.

# 3 Теоретическое введение

1. Процесс разработки программного обеспечения обычно разделяется на следующие этапы: – планирование, включающее сбор и анализ требований к функционалу и другим характеристикам разрабатываемого приложения; – проектирование, включающее в себя разработку базовых алгоритмов и спецификаций, определение языка программирования; – непосредственная разработка приложения: – кодирование — по сути создание исходного текста программы (возможно в нескольких вариантах); – анализ разработанного кода; – сборка, компиляция и разработка исполняемого модуля; – тестирование и отладка, сохранение произведённых изменений; – документирование. Для создания исходного текста программы разработчик может воспользоваться любым удобным для него редактором текста: vi, vim, mceditor, emacs, geany и др. После завершения написания исходного кода программы (возможно состоящей из нескольких файлов), необходимо её скомпилировать и получить исполняемый модуль.
2. Стандартным средством для компиляции программ в ОС типа UNIX является GCC (GNU Compiler Collection). Это набор компиляторов для разного рода языков программирова- ния (С, C++, Java, Фортран и др.). Работа с GCC производится при помощи одноимённой управляющей программы gcc, которая интерпретирует аргументы командной строки, определяет и осуществляет запуск нужного компилятора для входного файла.

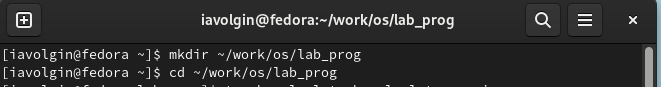
Некоторые опции компиляции в gcc Опция | Описание -c | компиляция без компоновки — создаются объектные файлы file.o -o | file-name задать имя file-name создаваемому файлу -g | поместить в файл (объектный или исполняемый) отладочную информацию для отладчика gdb -MM | вывести зависимости от заголовочных файлов C и/или C++ программ в формате, подходящем для утилиты make; при этом объектные или исполняемые файлы не будет созданы -Wall | вывод на экран сообщений об ошибках, возникших во время компиляции

1. Во время работы над кодом программы программист неизбежно сталкивается с появлением ошибок в ней. Использование отладчика для поиска и устранения ошибок в программе существенно облегчает жизнь программиста. В комплект программ GNU для ОС типа UNIX входит отладчик GDB (GNU Debugger)

Некоторые команды gdb Команда | Описание действия backtrace | вывод на экран пути к текущей точке останова (по сути вывод названий всех функций) break | установить точку останова (в качестве параметра может быть указан номер строки или название функции) clear | удалить все точки останова в функции continue | продолжить выполнение программы delete | удалить точку останова display | добавить выражение в список выражений, значения которых отображаются при достижении точки останова программы finish | выполнить программу до момента выхода из функции info breakpoints | вывести на экран список используемых точек останова info watchpoints | вывести на экран список используемых контрольных выражений list | вывести на экран исходный код (в качестве параметра может быть указано название файла и через двоеточие номера начальной и конечной строк) next | выполнить программу пошагово, но без выполнения вызываемых в программе функций print | вывести значение указываемого в качестве параметра выражения run | запуск программы на выполнение set | установить новое значение переменной step | пошаговое выполнение программы watch | установить контрольное выражение, при изменении значения которого программа будет остановлена

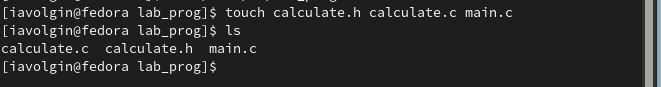
# 4 Выполнение лабораторной работы

Сперва нужно было создать нужный подкаталог ~/work/os/lab\_prog (рис. ??).

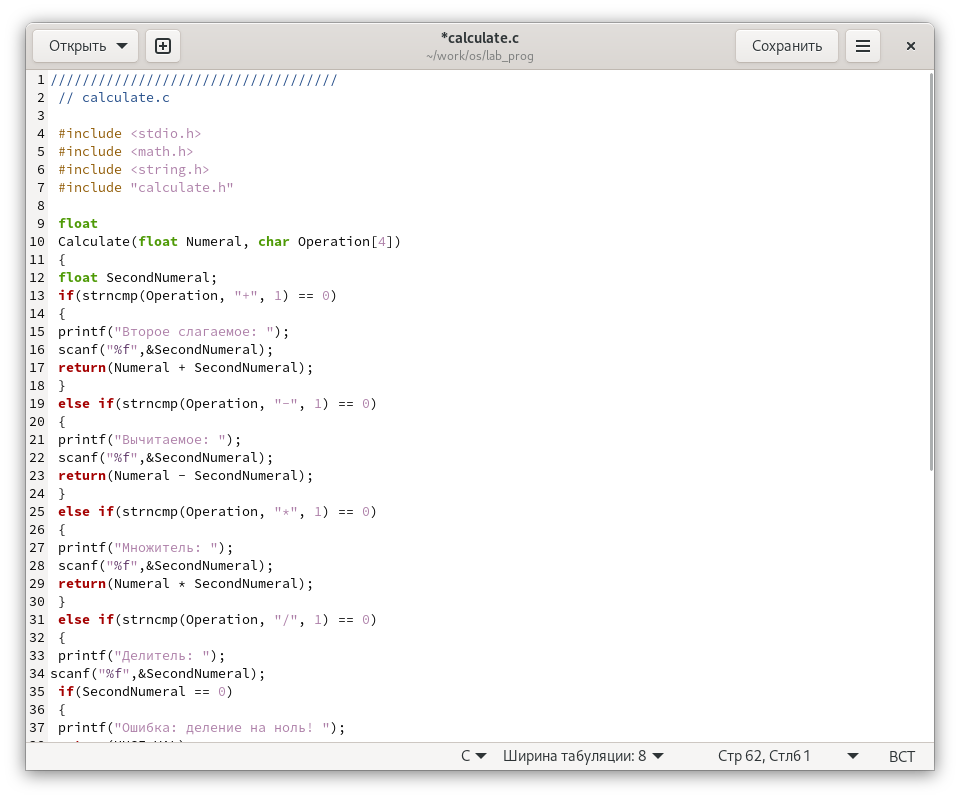


Создание подкаталога

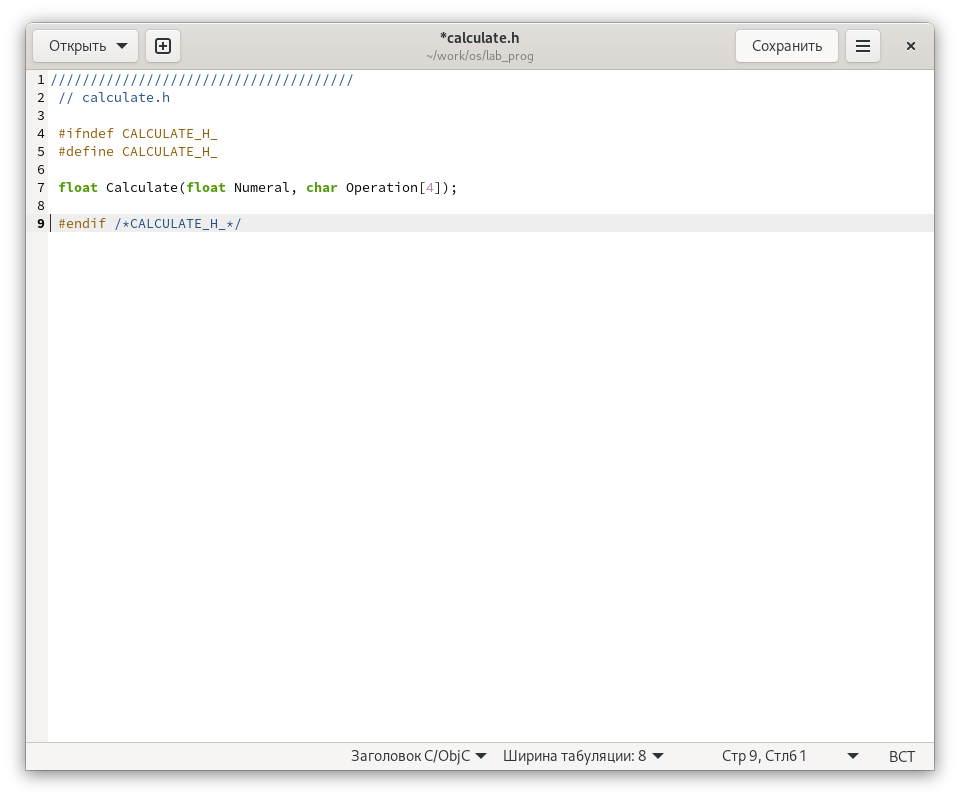
Затем создать в нем файлы calculate.h, calculate.c, main.c (рис. ??) и ввести код в эти файлы: calculate.с (рис. ??), calculate.р (рис. ??), main.c (рис. ??)



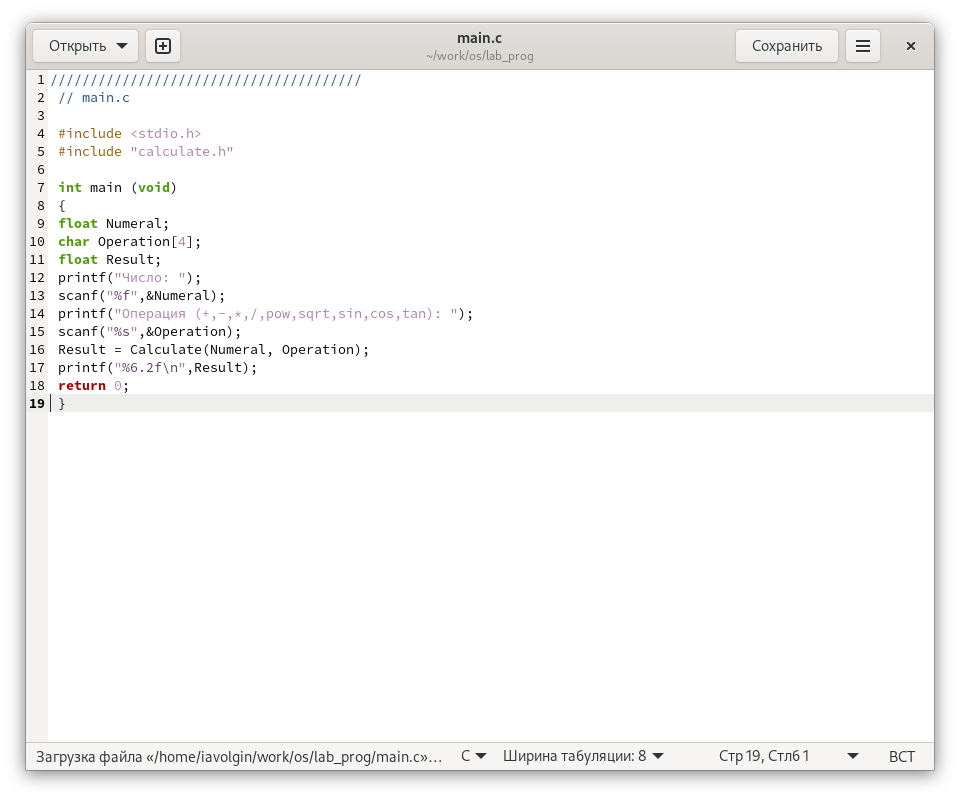
Создание фалов



Код calculate.с

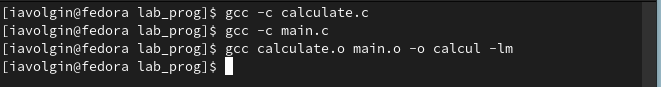


Код calculate.h



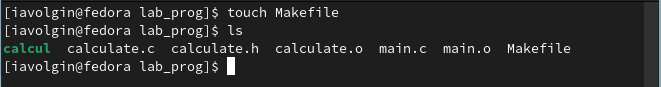
Код main.с

Далее нужно было выполнить компиляцию программы посредством gcc (рис. ??).

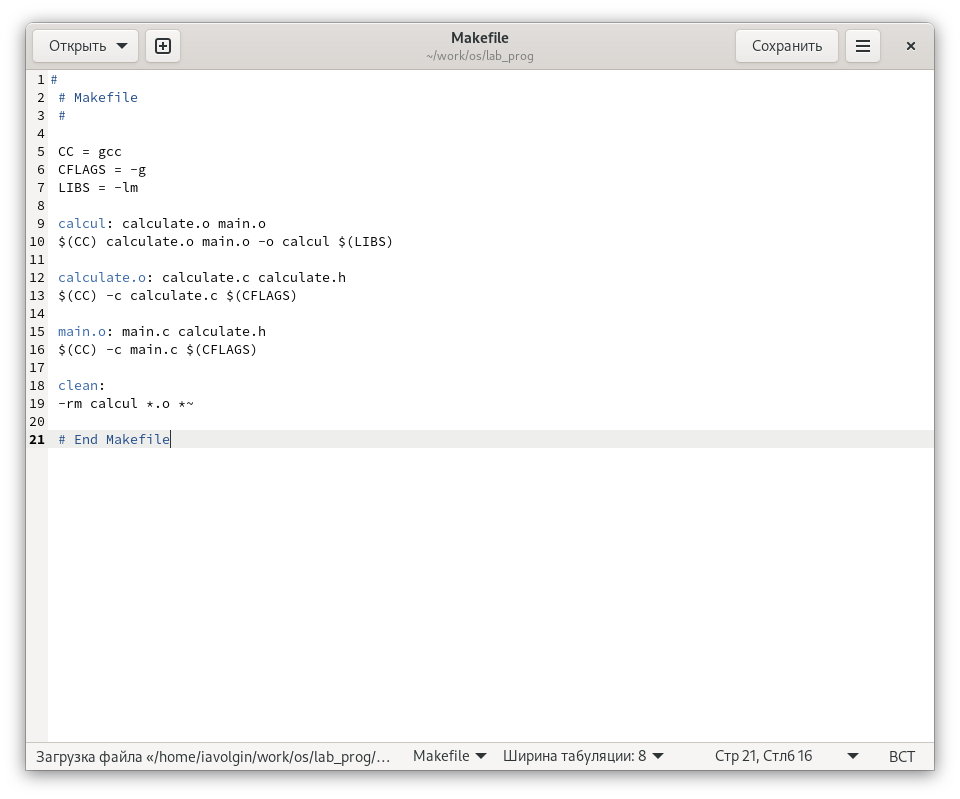


Компиляция программы

Затем нужно было проверить код на синтаксические ошибки, я их не нашел. После этого создать Makefile (рис. ??) и написать там следующий код (рис. ??).

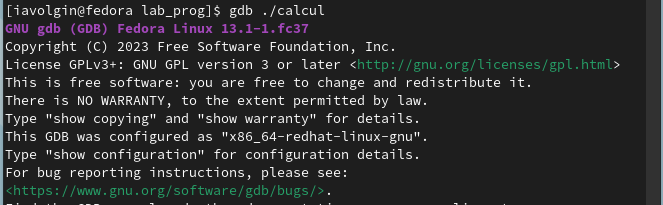


Создание Makefile

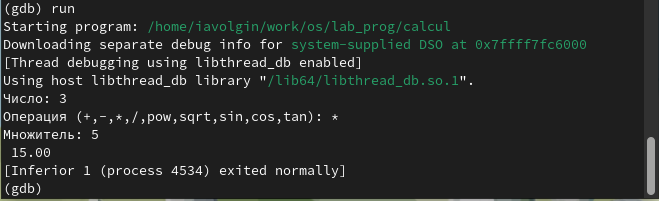


Код Makefile

Далее я запустил gdb, загрузив в него программ отладки (рис. ??) и затем запустил саму программу уже внутри отладчика (рис. ??).

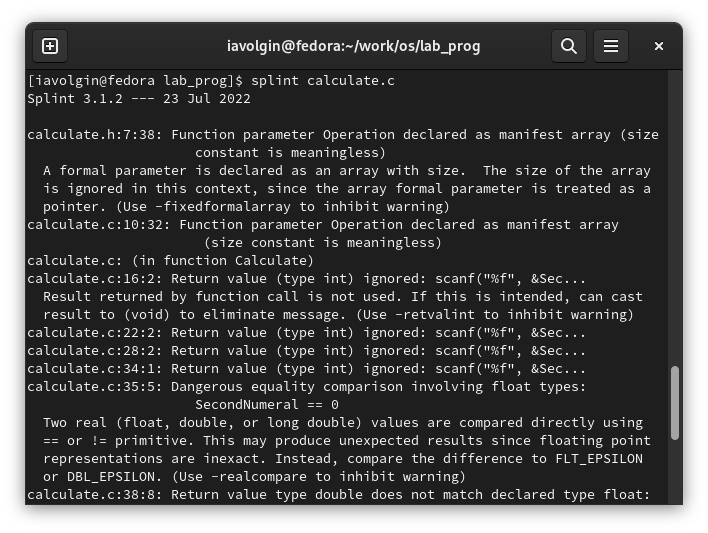


Запускаем gdb

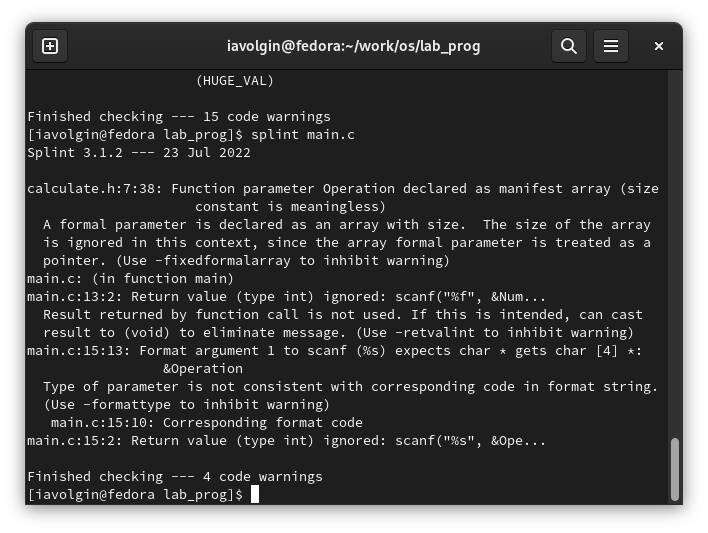


Запускаем программу внутри отладчика

После этого я пытался выполнить следующие задания и у меня никак не получалось и я не мог найти в чем проблема, поэтому я перешел к последнему пункту, где с помощью утилиты splint нужно было проанализировать коды файлов calculate.c (рис. ??) и main.c (рис. ??).



Анализ кода calculate.c



Анализ кода main.c

# 5 Выводы

В ходе выполнения данной лабораторной работы я приобрел простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.