Лабораторная работа №13

Операционные системы

Сабралиева Марворид Нуралиевна

Содержание

# 1 Цель работы

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

# 2 Выполнение лабораторной работы

1. В домашнем каталоге создадим подкаталог ~/work/os/lab\_prog. (рис. [1](#fig:001)).

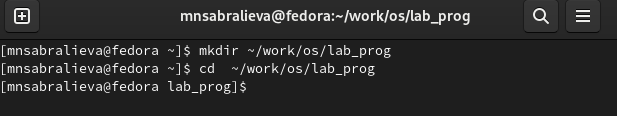


Figure 1: Создание подкаталога

1. Создадим 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.(рис. [2](#fig:002)).

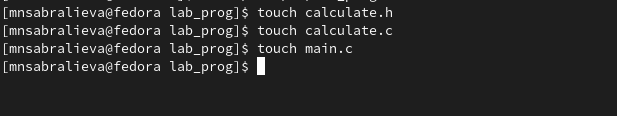


Figure 2: Создание файлов

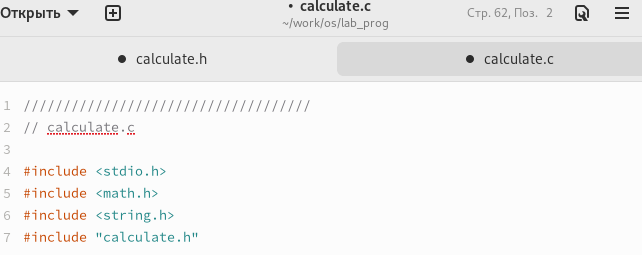


Figure 3: Реализация функций калькулятора в файле calculate.h

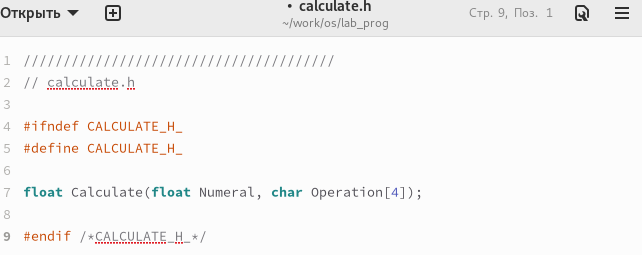


Figure 4: Интерфейсный файл calculate.h, описывающий формат вызова функции калькулятора



Figure 5: Основной файл main.c, реализующий интерфейс пользователя к калькулятору

1. Выполните компиляцию программы посредством gcc (рис. [6](#fig:006)).

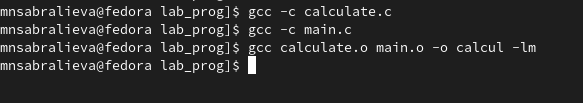


Figure 6: компиляция программы

1. Создадим Makefile и с помощью gdb выполним отладку программы calcul (рис. [7](#fig:007)).

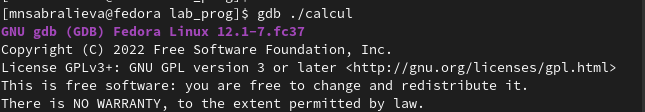


Figure 7: запустим отладчик

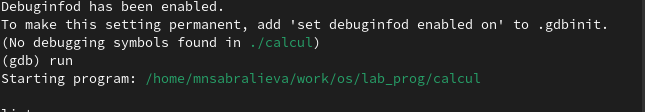


Figure 8: запуск программы

1. С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c. (рис. [9](#fig:009)).

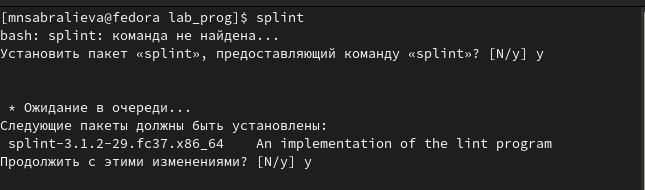


Figure 9: утилита splint

# 3 Выводы

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

# 4 Контрольные вопросы

1. Как получить информацию о возможностях программ gcc, make, gdb и др.? Ответ: Для этого есть команда man и предлагающиеся к ней файлы.
2. Назовите и дайте краткую характеристику основным этапам разработки прило- жений в UNIX. Ответ: Кодировка, Компиляция, Тест.
3. Что такое суффикс в контексте языка программирования? Приведите примеры использования. Ответ: Это расширения файлов.
4. Каково основное назначение компилятора языка С в UNIX? Ответ: Программа gcc, которая интерпретирует к определенному языку программирования аргументы командной строки и определяет запуск нужного компилятора для нужного файла.
5. Для чего предназначена утилита make? Ответ: Для компиляции группы файлов. Собрания из них программы, и последующего удаления.
6. Приведите пример структуры Makefile. Дайте характеристику основным элементам этого файла. Ответ: program: main.o lib.o cc -o program main.o lib.o main.o lib.o: defines.h В имени второй цели указаны два файла и для этой же цели не указана команда компиляции. Кроме того, нигде явно неуказана зависимость объектных файлов от «\*.c»-файлов. Дело в том, что программа make имеет предопределённые правила для получения файлов с определёнными расширениями. Так, для цели-объектного файла (расширение «.o») при обнаружении соответствующего файла с расширением «.c» будет вызван компилятор «сс -с» с указанием в параметрах этого «.c»-файла и всех файлов-зависимостей.
7. Назовите основное свойство, присущее всем программам отладки. Что необходимо сделать, чтобы его можно было использовать? Ответ: Программы для отладки нужны для нахождения ошибок в программе. Для их использования надо скомпилировать программу таким образом, чтобы отладочная информация содержалась в конечном бинарном файле.
8. Назовите и дайте основную характеристику основным командам отладчика gdb. Ответ: backtrace – выводит весь путь к текущей точке останова, то есть названия всех функций, начиная от main(); иными словами, выводит весь стек функций; break – устанавливает точку останова; параметром может быть номер строки или название функции; clear – удаляет все точки останова на текущем уровне стека (то есть в текущей функции); continue – продолжает выполнение программы от текущей точки до конца; delete – удаляет точку останова или контрольное выражение; display – добавляет выражение в список выражений, значения которых отображаются каждый раз при остановке программы; finish – выполняет программу до выхода из текущей функции; отображает возвращаемое значение,если такое имеется; info breakpoints – выводит список всех имеющихся точек останова; info watchpoints – выводит список всех имеющихся контрольных выражений; list – выводит исходный код; в качестве параметра передаются название файла исходного кода, затем, через двоеточие, номер начальной и конечной строки; next – пошаговое выполнение программы, но, в отличие от команды step, не выполняет пошагово вызываемые функции; print – выводит значение какого-либо выражения (выражение передаётся в качестве параметра); run – запускает программу на выполнение; set – устанавливает новое значение переменной step – пошаговое выполнение программы; watch – устанавливает контрольное выражение, программа остановится, как только значение контрольного выражения изменится;
9. Опишите по шагам схему отладки программы, которую Вы использовали при выполнении лабораторной работы. Ответ:
10. gdb –silent ./calcul
11. run
12. list
13. backtrace
14. breakpoints
15. print Numeral
16. Splint (Не использовался по причине отсутствия команды в консоли).
17. Прокомментируйте реакцию компилятора на синтаксические ошибки в программе при его первом запуске. Ответ: Консоль выводит ошибку с номером строки и ошибочным сегментом, но при этом есть возможность выполнить программу сразу.
18. Назовите основные средства, повышающие понимание исходного кода программы. Ответ:
19. Правильный синтаксис
20. Наличие комментариев
21. Разбиение большой сложной программы на несколько сегментов попроще.
22. Каковы основные задачи, решаемые программой splint? Ответ: split – разбиение файла на меньшие, определённого размера. Может разбивать текстовые файлы по строкам и любые – по байтам. По умолчанию читает со стандартного ввода и создает файлы с именами вида xaa, xab и т.д. По умолчанию разбиение идёт по 1000 строк в файле.

# Список литературы