Отчёт по лабораторной работе №14

Серегин Денис Алексеевич

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.

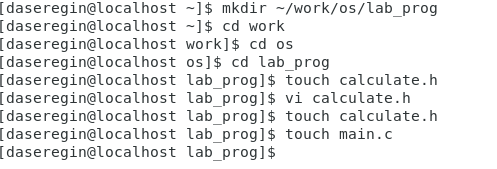
# Задания

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):

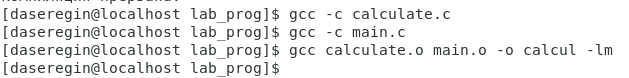
* – Запустите отладчик GDB, загрузив в него программу для отладки: gdb ./calcul
* – Для запуска программы внутри отладчика введите команду run: run
* – Для постраничного (по 9 строк) просмотра исходного код используйте команду list: list
* – Для просмотра строк с 12 по 15 основного файла используйте list с параметрами: list 12,15
* – Для просмотра определённых строк не основного файла используйте list с параметрами: list calculate.c:20,29
* – Установите точку останова в файле calculate.c на строке номер 21: list calculate.c:20,27 break 21
* – Выведите информацию об имеющихся в проекте точка останова: info breakpoints
* – Запустите программу внутри отладчика и убедитесь, что программа остановится в момент прохождения точки останова: run 5 - backtrace
* – Посмотрите, чему равно на этом этапе значение переменной Numeral, введя: print Numeral На экран должно быть выведено число 5.
* – Сравните с результатом вывода на экран после использования команды: display Numeral
* – Уберите точки останова: info breakpoints delete 1
  1. С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c

# Выполнение лабораторной работы

1. В домашнем каталоге создал подкаталог ~/work/os/lab\_prog. Создал 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 1. 1)
* 
* Figure 1: Создание файлов и директории

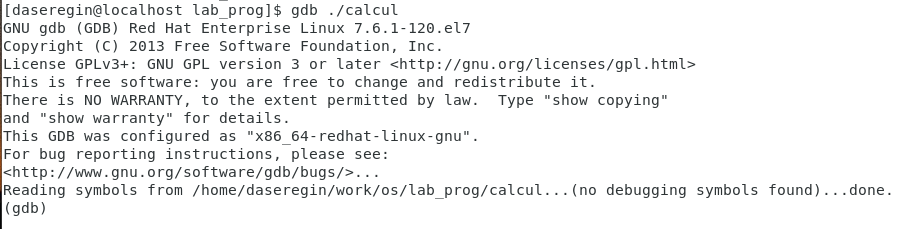
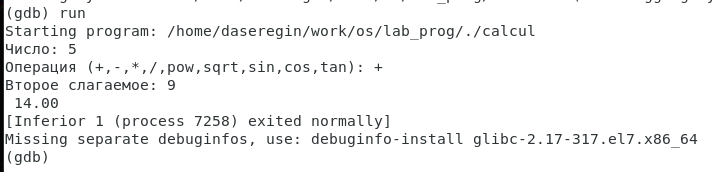
1. Выполнил компиляцию программы посредством gcc:

* gcc -c calculate.c
* gcc -c main.c
* gcc calculate.o main.o -o calcul –lm
* (рис 2. 2)
* 
* Figure 2: Компиляция программы с помощью gcc

1. Исправил синтаксические ошибки:

* scanf(“%s”,Operation)

1. Создал Makefile с содержанием из указаний к лабораторной работе
2. С помощью gdb выполнил отладку программы calcul (перед использованием gdb исправил Makefile: CFLAGS=-g)

* – Запустил отладчик GDB, загрузив в него программу для отладки: gdb ./calcul
* (рис 3. 3)
* 
* Figure 3: Запуск калькулятора в отладчике GDB
* – Для запуска программы внутри отладчика ввел команду run
* (рис 4. 4)
* 
* Figure 4: Работа команды run

1. С помощью утилиты splint попробовал проанализировать коды файлов calculate.c и main.c.

(рис 5. 5)

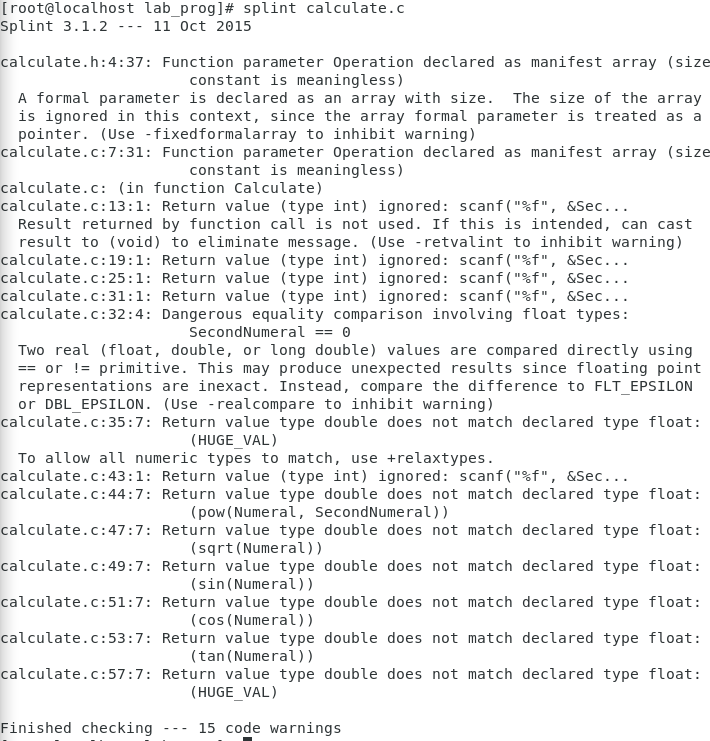


Figure 5: Работа команды splint

(рис 6. 6)

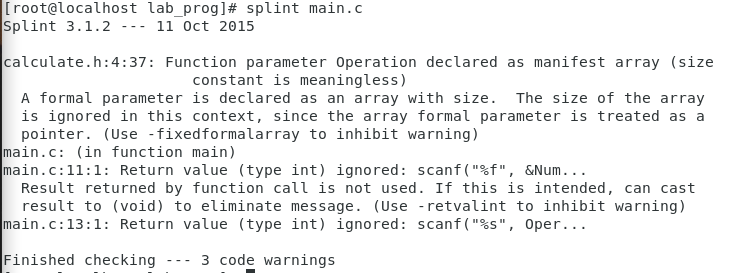


Figure 6: Работа команды splint

# Вывод

В ходе лабораторной работы мне удалось 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.

# Использованные ресурсы

https://onstartup.ru/razrabotka/splint/