Отчет

Лабораторная работа №13

Иван Подоляк НПМбд-02-21

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. 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.

* Для запуска программы внутри отладчика введите команду run.
* Постранично (по 9 строк) просмотрите исходный код.
* Для просмотра строк с 12 по 15 основного файла используйте list с параметрами.
* Для просмотра определённых строк не основного файла используйте list с параметрами.
* Установите точку остановки в файле calculate.c на строке номер 21.
* Выведите информацию об имеющихся в проекте точках остановок.
* Запустите программу внутри отладчика и убедитесь, что программа остановится в момент прохождения точки остановки.
* Посмотрите, чему равно на этом этапе значение переменной Numeral.
* Сравните с результатом вывода на экран.
* Уберите точки остановок.

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

# Теоретическое введение

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.

# Выполнение лабораторной работы

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

создание подкаталога

создание подкаталога

1. Создаю в нём файлы: *calculate.h*, *calculate.c*, *main.c*. (рис. [-@fig:002]) Пишу код для работы программы в созданных файлах. (рис. [-@fig:003]) (рис. [-@fig:004]) (рис. [-@fig:005]) (рис. [-@fig:006])

создание файлов

создание файлов

calculate.c

calculate.c

calculate.c

calculate.c

calculate.h

calculate.h

main.c

main.c

1. Выполняю компиляцию программы посредством gcc.(рис. [-@fig:007]) Ошибок не нашлось.

компиляция программы

компиляция программы

1. Создаю Makefile. (рис. [-@fig:008]) Пишу в файле текст, указанный в лабораторной работе. (рис. [-@fig:009]) Меняю текст, чтобы я могла работать с отладчиком. (рис. [-@fig:010]) Запускаю Makefile. (рис. [-@fig:011])

создание файла

создание файла

создание Makefile

создание Makefile

изменение файла

изменение файла

makefile

makefile

1. С помощью gdb выполняю отладку программы calcul: Запускаю отладчик GDB, загрузив в него программу для отладки. (рис. [-@fig:012])

* Для запуска программы внутри отладчика ввожу команду run. (рис. [-@fig:013])
* Постранично (по 9 строк) просматриваю исходный код. (рис. [-@fig:014])
* Для просмотра строк с 12 по 15 основного файла использую list с параметрами 12,15. (рис. [-@fig:015])
* Для просмотра определённых строк не основного файла использую list с параметрами имя\_файла: 20,29. (рис. [-@fig:016])
* Устанавливаю точку остановки в файле calculate.c на строке номер 21. (рис. [-@fig:017])
* Вывожу информацию об имеющихся в проекте точках остановок. (рис. [-@fig:018])
* Запускаю программу внутри отладчика и убеждаюсь, что программа останавливается в момент прохождения точки остановки. (рис. [-@fig:019])
* Посматриваю, чему равно на этом этапе значение переменной Numeral и сравниваю с результатом вывода на экран. (рис. [-@fig:020])
* Убираю точки остановок.(рис. [-@fig:021])

запуск отладчика

запуск отладчика

запуск программы

запуск программы

постраничный просмотр

постраничный просмотр

просмотр строк 12-15

просмотр строк 12-15

просмотр строк не основного файла

просмотр строк не основного файла

установка точки остановки

установка точки остановки

вывод информации о точках остановок

вывод информации о точках остановок

запуск внутри отладчика

запуск внутри отладчика

сравнение значений Numeral

сравнение значений Numeral

удаление точек остановок

удаление точек остановок

1. С помощью утилиты splint анализирую коды файлов calculate.c и main.c.(рис. [-@fig:022])(рис. [-@fig:023])

calculate.c

calculate.c

main.c

main.c

# Выводы

Выполняя данную лабораторную работу я приобрёл простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.

# Список литературы