Лабораторная работа №13. Средства, применяемые при разработке программного обеспечения в ОС типа UNIX/Linux.

Emil A. Samigullin

27 April, 2022 Moscow, Russian Federation

[1]RUDN University, Moscow, Russian Federation

Лабораторная работа №13

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

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

Автор: Смирнов-Мальцев Егор Дмитриевич

Москва, 2022

Цель работы

Цель работы

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

Задание

Задание

1. Написать приложение, выполняющее функции калькулятора на языке C.

Теоретическое введение

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;

Теоретическое введение

- 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

Выполнение лабораторной работы

1. Создал каталог ~/work/os/lab_prog.
2. Написал на C программы, выполняющие функции калькулятора.
3. Написал Makefile, компилирующий программы из предыдущего пункта и запустил утилиту make.

Выполнение лабораторной работы

Выполнение лабораторной работы

4. С помощью gdb выполнил отладку calcul:

   - Запустил отладчик.
   - Запустил программу в отладчике.
   - Просмотрел первые 9 строк исходного кода.
   - Просмотрел с 12 по 15 строки исходного кода.
   - Просмотрел несколько строк неосновного файла.
   - Установил точку останова на строке 21.
   - Вывел информацию о точках останова.
   - Еще раз запустил программу.
   - Проверил значение переменной Numeral 2 способами.
   - Убрал точки останова.

5. С помощью splint просмотрел коды файлов main.c и calculate.c.

Ответы на контрольные вопросы

Ответы на контрольные вопросы

1. Дополнительную информацию о этих программах можно получить с помощью утилиты man.
2. Unix поддерживает следующие основные этапы разработки приложений:

- создание исходного кода программы;
- представление в виде файла;
- сохранение различных вариантов исходного текста;
- анализ исходного текста;
- компиляция исходного текста и построение исполняемого модуля;
- тестирование и отладка;
- проверка кода на наличие ошибок
- сохранение всех изменений, выполняемых при тестировании и отладке.

Ответы на контрольные вопросы

Ответы на контрольные вопросы

3. Суффикс определяет какая компиляция требуется. Суффиксы и префиксы указывают тип объекта. Одно из полезных свойств компилятора Си — его способность по суффиксам определять типы файлов. По суффиксу .c компилятор распознает, что файл abcd.c должен компилироваться, а по суффиксу .o, что файл abcd.o является объектным модулем и для получения исполняемой программы необходимо выполнить редактирование связей. Простейший пример командной строки для компиляции программы abcd.c и построения исполняемого модуля abcd имеет вид: gcc -o abcd abcd.c. Некоторые проекты предпочитают показывать префиксы в начале текста изменений для старых (old) и новых (new) файлов. Опция – prefix может быть использована для установки такого префикса. Плюс к этому команда bzr diff -p1 выводит префиксы в форме которая подходит для команды patch -p1.

Ответы на контрольные вопросы

Ответы на контрольные вопросы

4. Основное назначение компилятора с языка Си заключается в компиляции всей программы в целом и получении исполняемого модуля.
5. Утилита make освобождает пользователя от такой ручной компиляции всех файлов и служит для документирования взаимосвязей между файлами. Описание взаимосвязей и соответствующих действий хранится в так называемом make-файле, который по умолчанию имеет имя makefile или Makefile.

Ответы на контрольные вопросы

Ответы на контрольные вопросы

6. makefile может иметь вид:

```
#
# Makefile
#
CC = gcc
CFLAGS =
LIBS = -lm
calcul: calculate.o main.o
gcc calculate.o main.o -o calcul $(LIBS)
calculate.o: calculate.c calculate.h
gcc -c calculate.c $(CFLAGS)
main.o: main.c calculate.h
gcc -c main.c $(CFLAGS)
clean:
-rm calcul *.o *~
```

Ответы на контрольные вопросы

Ответы на контрольные вопросы

7. Пошаговая отладка программ заключается в том, что выполняется один оператор программы и, затем контролируются те переменные, на которые должен был воздействовать данный оператор.
8. Основные команды gdb:

- 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;

Ответы на контрольные вопросы

Ответы на контрольные вопросы

9. Схема отладки программы которую мы использовали при выполнении лабораторной работы.
   9.1 Выполнили компиляцию программы
   9.2 Увидели ошибки в программе
   9.3 Открыли редактор и исправили программу
   9.4 Загрузили программу в отладчик gdb
   9.5 run — отладчик выполнил программу, мы ввели требуемые значения.
   9.6 программа завершена, gdb не видит ошибок.
10. При первом запуске программы с синтаксической ошибкой отладчику не понравился формат %s для &Operation, т.к %s — символьный формат, а значит необходим только Operation.

Ответы на контрольные вопросы

Ответы на контрольные вопросы

11. При работаете с исходным кодом, который не вами разрабатывался, назначение различных конструкций может быть не совсем понятным. Система разработки приложений UNIX предоставляет различные средства, повышающие понимание исходного кода. К ним относятся:

- cscope - исследование функций, содержащихся в программе;
- splint — критическая проверка программ, написанных на языке Си.

Ответы на контрольные вопросы

Ответы на контрольные вопросы

12. Основные задачи, решаемые программой slint:
    - 12.1 Проверка корректности задания аргументов всех использованных в программе функций, а также типов возвращаемых ими значений;
    - 12.2 Поиск фрагментов исходного текста, корректных с точки зрения синтаксиса языка Си, но малоэффективных с точки зрения их реализации или содержащих в себе семантические ошибки;
    - 12.3 Общая оценка мобильности пользовательской программы.

Выводы

Выводы

- Я научился разрабатывать и тестировать простейшие приложения.