Лабораторная работа № 13

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

Шулуужук Айраана Вячеславовна НПИбд-02-22

Содержание

Список иллюстраций

Список таблиц

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. Написать командный файл, реализующий упрощённый механизм семафоров. Ко- мандный файл должен в течение некоторого времени t1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени t2<>t1, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (> /dev/tty#, где # — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имелась возможность взаимодействия трёх и более процессов.

2. Реализовать команду man с помощью командного файла. Изучите содержимое ката- лога /usr/share/man/man1. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой less сразу же просмотрев содержимое справки. Командный файл должен получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде или сообщение об отсутствии справки, если соответствующего файла нет в каталоге man1.

3. Используя встроенную переменную $RANDOM, напишите командный файл, генерирую- щий случайную последовательность букв латинского алфавита.

Учтите, что $RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

3 Теоретическое введение

3.1 Этапы разработки приложений

Процесс разработки программного обеспечения обычно разделяется на следующие этапы:

– планирование, включающее сбор и анализ требований к функционалу и другим ха- рактеристикам 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.

4 Выполнение лабораторной работы

Создадим нужный каталог и необходимые файлы (рис. 4.1)

```
[avshuluuzhuk@fedora ~]$ cd work
[avshuluuzhuk@fedora work]$ cd os
[avshuluuzhuk@fedora os]$ mkdir lab_prog
[avshuluuzhuk@fedora os]$ cd lab_prog
[avshuluuzhuk@fedora lab_prog]$ touch calculate.h
[avshuluuzhuk@fedora lab_prog]$ touch calculate.c
[avshuluuzhuk@fedora lab_prog]$ touch main.c
[avshuluuzhuk@fedora lab_prog]$
```

Рис. 4.1: создание файлов и каталога

Внесем тексты программ в файлы (рис. 4.2) (рис. 4.3) (рис. 4.4)

```
////////////////////////////////////
// calculate.c

#include <stdio.h>
#include <math.h>
#include <string.h>
#include "calculate.h"

float
Calculate(float Numeral, char Operation[4])
{
        float SecondNumeral;
        if(strncmp(Operation, "+", 1) == 0)
        {
                printf("Второе слагаемое: ");
                scanf("%f",&SecondNumeral);
                return(Numeral + SecondNumeral);
        }
        else if(strncmp(Operation, "-", 1) == 0)
        {
                printf("Вычитаемое: ");
                scanf("%f",&SecondNumeral);
                return(Numeral - SecondNumeral);
        }
```

Рис. 4.2: calculate.c

```
////////////////////////////////////////
// calculate.h

#ifndef CALCULATE_H_
#define CALCULATE_H_

float Calculate(float Numeral, char Operation[4]);

#endif /*CALCULATE_H_*/
~
```

Рис. 4.3: calculate.h

```
////////////////////////////////////////
// main.c

#include <stdio.h>
#include "calculate.h"

int
main (void)
{
        float Numeral;
        char Operation[4];
        float Result;
        printf("Число: ");
        scanf("%f",&Numeral);
        printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
        scanf("%s",&Operation);
        Result = Calculate(Numeral, Operation);
        printf("%6.2f\n",Result);
        return 0;
}
~
```

Рис. 4.4: main.c

Выполним компиляцию программы посредством gcc (рис. 4.5)

```
[avshuluuzhuk@fedora lab_prog1]$ gcc -c calculate.c
[avshuluuzhuk@fedora lab_prog1]$ gcc -c main.c
[avshuluuzhuk@fedora lab_prog1]$ gcc calculate.o main.o -o calcul -lm
[avshuluuzhuk@fedora lab_prog1]$ ls
calcul       calculate.c~  calculate.h~  main.c   main.o
calculate.c  calculate.h   calculate.o   main.c~
[avshuluuzhuk@fedora lab_prog1]$
```

Рис. 4.5: компиляция программы

Создадим файл Makefile со следующим содержащим (рис. 4.6)

```
#
# 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)

15 main.o: main.c calculate.h
	gcc -c main.c $(CFLAGS)
clean:
	-rm calcul *.o *~

# End Makefile
```

Рис. 4.6: Makefile

С помощью gdb выполним отладку программы calcul. Запустим отладчик GDB, загрузив в него программу для отладки и введем команду run (рис. 4.7)

```
(gdb) run
Starting program: /home/avshuluuzhuk/work/os/lab_prog1/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 3
  8.00
[Inferior 1 (process 4867) exited normally]
(gdb)
```

Рис. 4.7: скрипт 3

С помощью утилиты splint проанализируем коды файлов calculate.c и main.c (рис. 4.8) (рис. 4.9)

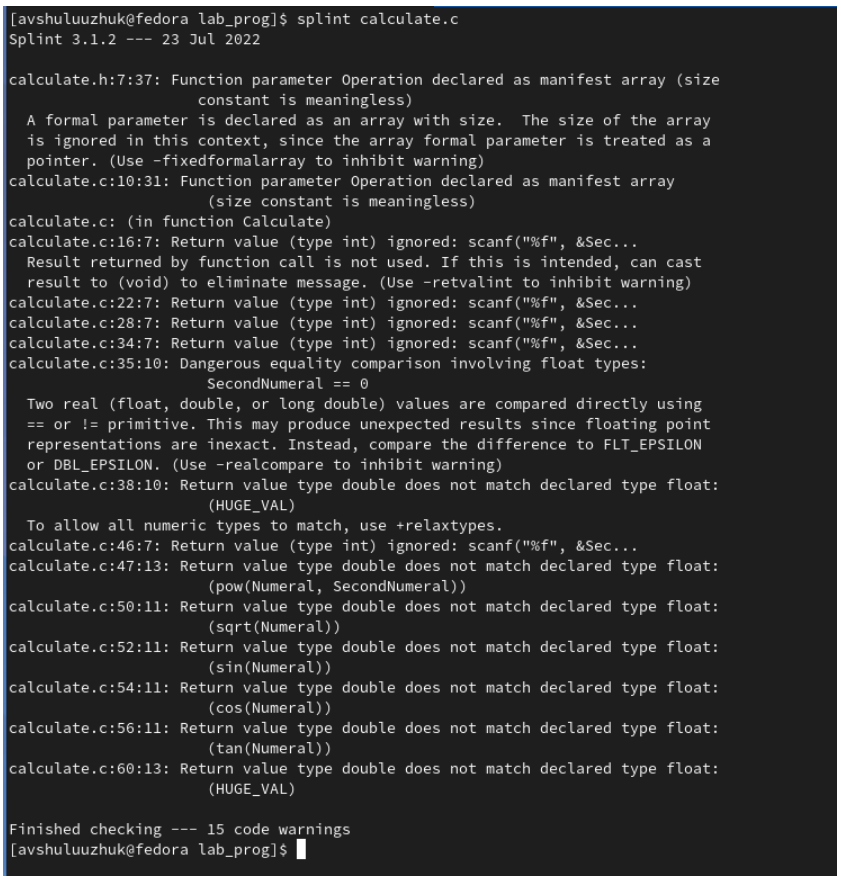

Рис. 4.8: анализ файла calculate.c

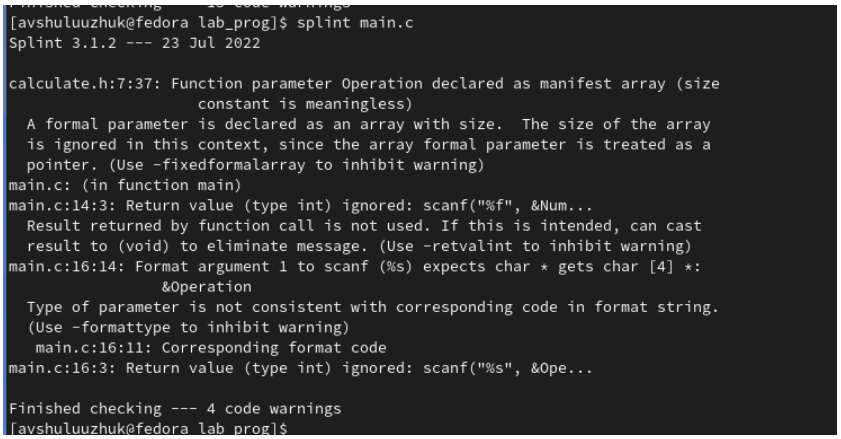

Рис. 4.9: анализ файла main.c

5 Выводы

В ходе выполнения работы мы приобрели простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования C калькулятора с простейшими функциями.