Лабораторная работа №13

Операционные системы

Канева Е. П.

4 мая 2023

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Канева Екатерина Павловна
- Студентка группы НКАбд-02-22
- Российский университет дружбы народов
- https://github.com/Nevseros/study_2022-2023_os-intro

Вводная часть

Цель

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.

Задания

1. Написать программные файлы.
2. Скомпилировать их с помощью `gcc`.
3. Отладить их с помощью `gdb`.
4. Проанализировать код с помощью утилиты `splint`.

Выполнение работы

Создание файлов

Создала нужные каталог и файлы (рис. 1):

```
[epkaneva@epkaneva ~]$ mkdir -p work/os/lab_prog
[epkaneva@epkaneva ~]$ cd work/os/lab_prog
[epkaneva@epkaneva lab_prog]$ gedit calculate.h calculate.c main.c &
```

Figure 1: Создание каталога и файлов.

Текст файлов

Далее перенесла текст программных файлов и Makefile (рис. 2, 3, 4 и 5):

```
1 ////////////////////////////////////
2 // calculate.c
3
4 #include <stdio.h>
5 #include <math.h>
6 #include <string.h>
7 #include "calculate.h"
8
9 float
10 Calculate(float Numeral, char Operation[4])
11 {
12     float SecondNumeral;
13     if(strncmp(Operation, "+", 1) == 0)
14     {
15         printf("Второе слагаемое: ");
16         scanf("%f",&SecondNumeral);
17         return(Numeral + SecondNumeral);
18     }
```

Figure 2: Файл calculate.c.

Текст файлов

```
1 //////////////////////////////////////
2 // calculate.h
3
4 #ifndef CALCULATE_H_
5 #define CALCULATE_H_
6
7 float Calculate(float Numeral, char Operation[4]);
8
9 #endif /*CALCULATE_H_*/
```

Figure 3: Файл calculate.h.

Текст файлов

```c
////////////////////////////////////////
// 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;
}
```

Figure 4: Файл main.c.

Текст файлов

```
#
# 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 *~

# End Makefile
```

Figure 5: Makefile.

Компиляция

Выполнила компиляцию программы (рис. 6):

```
[epkaneva@epkaneva lab_prog]$ gcc -c calculate.c
[epkaneva@epkaneva lab_prog]$ gcc -c main.c
[epkaneva@epkaneva lab_prog]$ gcc calculate.o main.o -o calcul -lm
```

Figure 6: Компиляция программы.

Запуск отладчика

Запустила отладчик GDB (рис. 7):

```
[epkaneva@epkaneva lab_prog]$ gdb ./calcul
GNU gdb (GDB) Fedora 12.1-2.fc36
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./calcul...

This GDB supports auto-downloading debuginfo from the following URLs:
https://debuginfod.fedoraproject.org/
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
(No debugging symbols found in ./calcul)
```

Figure 7: Запуск отладчика.

Запуск программы

В отладчике с помощью команды `run` запустила программу (рис. 8):

```
Число: 7
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): sin
  0.66
[Inferior 1 (process 3273) exited normally]
```

Figure 8: Запуск программы.

Добавление брейкпоинтов

Добавила брейкпоинт и проверила его наличие с помощью команды `info breakpoints` (рис. 9):

```
(gdb) info breakpoints
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   <PENDING>  calculate.c:21
```

Figure 9: Проверка наличия брейкпоинта.

Удаление брейкпоинта

Удалила брейкпоинт с помощью команды `delete 1` (рис. 10):

```
(gdb) delete 1
(gdb) info breakpoints
No breakpoints or watchpoints.
```

Figure 10: Удаление брейкпоинта.

Анализ кода

С помощью утилиты `splint` проанализировали коды файлов `calculate.c` (рис. 11) и `main.c` (рис. 12):

```
calculate.h:7:37: Function parameter Operation declared as manifest array (size
                    constant is meaningless)
  A formal parameter is declared as an array with size.  The size of the array
  is ignored in this context, since the array formal parameter is treated as a
  pointer. (Use -fixedformalarray to inhibit warning)
calculate.c:10:31: Function parameter Operation declared as manifest array
                     (size constant is meaningless)
calculate.c: (in function Calculate)
calculate.c:16:9: Return value (type int) ignored: scanf("%f", &Sec...
  Result returned by function call is not used. If this is intended, can cast
  result to (void) to eliminate message. (Use -retvalint to inhibit warning)
calculate.c:22:9: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:9: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:9: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:12: Dangerous equality comparison involving float types:
                     SecondNumeral == 0
  Two real (float, double, or long double) values are compared directly using
  == or != primitive. This may produce unexpected results since floating point
  representations are inexact. Instead, compare the difference to FLT_EPSILON
  or DBL_EPSILON. (Use -realcompare to inhibit warning)
calculate.c:38:19: Return value type double does not match declared type float:
                     (HUGE_VAL)
  To allow all numeric types to match, use +relaxtypes.
calculate.c:45:9: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:46:15: Return value type double does not match declared type float:
                     (pow(Numeral, SecondNumeral))
calculate.c:48:50: Return value type double does not match declared type float:
                     (sqrt(Numeral))
calculate.c:49:49: Return value type double does not match declared type float:
                     (sin(Numeral))
calculate.c:50:49: Return value type double does not match declared type float:
                     (cos(Numeral))
```

Анализ кода

```
calculate.h:7:37: Function parameter Operation declared as manifest array (size
                    constant is meaningless)
  A formal parameter is declared as an array with size.  The size of the array
  is ignored in this context, since the array formal parameter is treated as a
  pointer. (Use -fixedformalarray to inhibit warning)
main.c: (in function main)
main.c:14:5: Return value (type int) ignored: scanf("%f", &Num...
  Result returned by function call is not used. If this is intended, can cast
  result to (void) to eliminate message. (Use -retvalint to inhibit warning)
main.c:16:16: Format argument 1 to scanf (%s) expects char * gets char [4] *:
                &Operation
  Type of parameter is not consistent with corresponding code in format string.
  (Use -formattype to inhibit warning)
   main.c:16:13: Corresponding format code
main.c:16:5: Return value (type int) ignored: scanf("%s", &Ope...

Finished checking --- 4 code warnings
```

Figure 12: `main.c`.

Заключение

Выводы

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