

Лабораторная работа №13

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

Рогожина Н.А.

14 апреля 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Рогожина Надежда Александровна
- Студентка 1го курса, НКАбд-02-22
- Компьютерные и информационные науки
- Российский университет дружбы народов
- Github

Вводная часть

- 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.

Содержание лабораторной работы

Стандартным средством для компиляции программ в ОС типа UNIX является GCC (GNU Compiler Collection). Это набор компиляторов для разного рода языков программирования (C, C++, Java, Фортран и др.). Работа с GCC производится при помощи одноимённой управляющей программы **gcc**, которая интерпретирует аргументы командной строки, определяет и осуществляет запуск нужного компилятора для входного файла.

Задание

Задачи:

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`:

```
gcc -c -g calculate.c
```

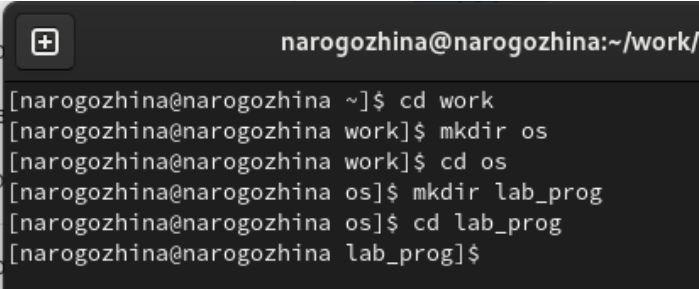
```
gcc -c -g main.c
```

```
gcc calculate.o main.o -o calcul -lm
```

4. При необходимости исправьте синтаксические ошибки.

5. Создайте Makefile.
6. С помощью gdb выполните отладку программы calcul (перед использованием gdb исправьте Makefile):
7. С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c.

Выполнение

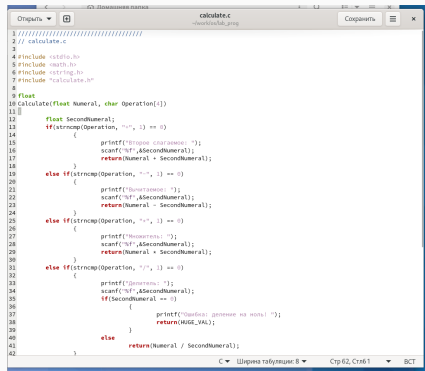


A terminal window with a dark background and a title bar. The title bar contains a plus icon in a square and the text 'narogozhina@narogozhina:~/work/'. The terminal shows a sequence of commands and their outputs:

```
[narogozhina@narogozhina ~]$ cd work
[narogozhina@narogozhina work]$ mkdir os
[narogozhina@narogozhina work]$ cd os
[narogozhina@narogozhina os]$ mkdir lab_prog
[narogozhina@narogozhina os]$ cd lab_prog
[narogozhina@narogozhina lab_prog]$
```

On the left side of the terminal window, there is a vertical list of text labels: 'Из', 'М', 'Кор', and 'VBo'.

Рис. 1: Создание подкаталога



```

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     }
19     else if(strncmp(Operation, "-", 1) == 0)
20     {
21         printf("Выващаемое: ");
22         scanf("%f", &SecondNumeral);
23         return(Numeral - SecondNumeral);
24     }
25     else if(strncmp(Operation, "*", 1) == 0)
26     {
27         printf("Умножаем: ");
28         scanf("%f", &SecondNumeral);
29         return(Numeral * SecondNumeral);
30     }
31     else if(strncmp(Operation, "/", 1) == 0)
32     {
33         printf("Делим: ");
34         scanf("%f", &SecondNumeral);
35         if(SecondNumeral == 0)
36         {
37             printf("Ошибка: деление на ноль!\n");
38             return(HUGE_VAL);
39         }
40         else
41             return(Numeral / SecondNumeral);
42     }
43 }

```

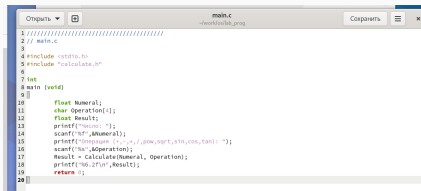
Рис. 2: calculate.c



The screenshot shows a code editor window with the title bar "calculate.h" and a subtitle "c:\work\calc\prog". The editor contains the following C code:

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

Рис. 3: calculate.h



The image shows a code editor window with the title bar 'main.c' and a path '-/work/sulab_prog'. The editor contains the following C code:

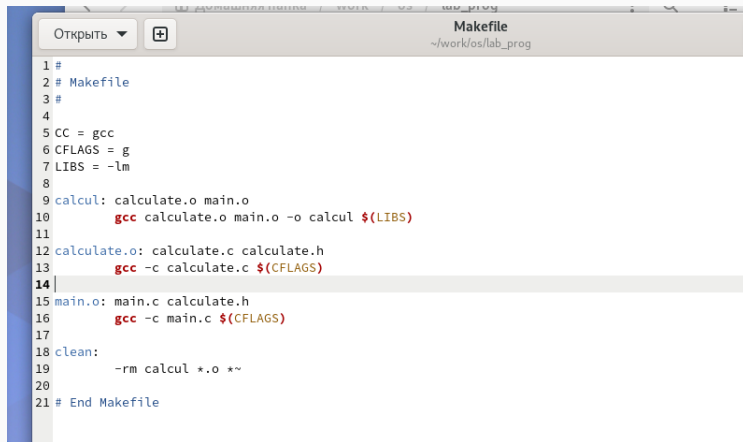
```
1 //////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
5 #include "calculate.h"
6
7 int
8 main (void)
9 {
10     float Numeral;
11     char Operation[4];
12     float Result;
13     printf("numenc: ");
14     scanf("%f", &Numeral);
15     printf("Operation (<+,-,*,/,pow,sqrt,sin,cos,tan>: ");
16     scanf("%s", &Operation);
17     Result = Calculate(Numeral, Operation);
18     printf("exp.2Fun", Result);
19     return 0;
20 }
```

Рис. 4: main.c

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

Рис. 5: Компиляция программы

Makefile



The screenshot shows a text editor window with the title bar 'Makefile' and a subtitle '~/.work/os/lab_prog'. The editor contains a Makefile script with the following content:

```
1 #
2 # Makefile
3 #
4
5 CC = gcc
6 CFLAGS = g
7 LIBS = -lm
8
9 calcul: calculate.o main.o
10     gcc calculate.o main.o -o calcul $(LIBS)
11
12 calculate.o: calculate.c calculate.h
13     gcc -c calculate.c $(CFLAGS)
14
15 main.o: main.c calculate.h
16     gcc -c main.c $(CFLAGS)
17
18 clean:
19     -rm calcul *.o *~
20
21 # End Makefile
```

Рис. 6: Создание Makefile

Запуск отладчика и проверка программы

```
make: «calcul» не требует обновления.
[narogozhina@narogozhina lab_prog]$ gdb ./calcul
GNU gdb (GDB) Fedora Linux 13.1-2.fc37
Copyright (C) 2023 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...
(gdb) run
Starting program: /home/narogozhina/work/os/lab_prog/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.
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Множитель: 15
75.00
[Inferior 1 (process 3685) exited normally]
(gdb)
```

Рис. 7: Запуск gdb

```
75.00  
[Inferior 1 (process 3685) exited normally]  
(gdb) list  
1      ///////////////////////////////////////////  
2      // main.c  
3  
4      #include <stdio.h>  
5      #include "calculate.h"  
6  
7      int  
8      main (void)  
9      {  
10         float Numeral;  
(gdb)
```

Рис. 8: list

Просмотр определённых строк не основного файла

```
14         scanf("%f",&Numeral);
15         printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
(gdb) list calculate.c:20,29
20         {
21             printf("Вычитаемое: ");
22             scanf("%f",&SecondNumeral);
23             return(Numeral - SecondNumeral);
24         }
25     else if(strncmp(Operation, "*", 1) == 0)
26     {
27         printf("Множитель: ");
28         scanf("%f",&SecondNumeral);
29         return(Numeral * SecondNumeral);
(gdb)
```

Рис. 9: list calculate.c:20,29

Установка точки остановки в файле calculate.c на строке номер 21

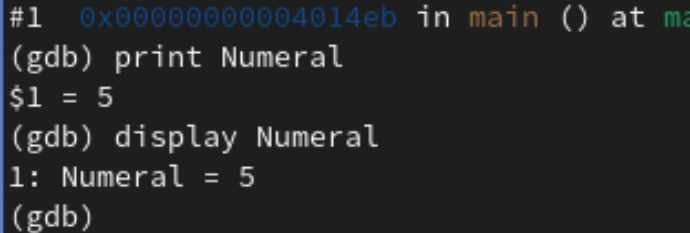
```
28         scanf("%i",&SecondNumeral);
29         return(Numeral * SecondNumeral);
(gdb) list calculate.c:20,27
20     {
21         printf("Вычитаемое: ");
22         scanf("%f",&SecondNumeral);
23         return(Numeral - SecondNumeral);
24     }
25     else if(strncmp(Operation, "*", 1) == 0)
26     {
27         printf("Множитель: ");
(gdb) break 21
Breakpoint 1 at 0x40120f: file calculate.c, line 21.
(gdb)
```

Рис. 10: Точка остановки

```
(gdb) run
Starting program: /home/narogozhina/work/os/lab_prog/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): -

Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffffdea4 "-") at calculate.c:21
21                                     printf("Вычитаемое: ");
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffffdea4 "-") at calculate.c:21
#1 0x00000000004014eb in main () at main.c:17
(gdb)
```

Рис. 11: run + backtrace



```
#1  0x00000000004014eb in main () at ma
(gdb) print Numeral
$1 = 5
(gdb) display Numeral
1: Numeral = 5
(gdb)
```

Рис. 12: print/display Numeral

```
(gdb) display Numeral
1: Numeral = 5
(gdb) info breakpoints
Num      Type             Disp Enb Address              What
1        breakpoint      keep y   0x00000000000040120f in Calculate at calculate.c:21
          breakpoint already hit 1 time
(gdb) delete 1
(gdb)
```

Рис. 13: Удаление точек остановки


```
1 breakpoint keep y 0x00000000401201 in calculate
breakpoint already hit 1 time
(gdb) delete 1
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/narogozhina/work/os/lab_prog/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): -
Вычитаемое: 1
4.00
[Inferior 1 (process 3796) exited normally]
(gdb) █
```

Рис. 14: run

```
[narogozhina@narogozhina lab_prog]$ splint calculate.c
Splint 3.1.2 --- 23 Jul 2022

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:16:4: 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:22:4: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:4: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:4: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:7: 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:38:12: Return value type double does not match declared type float:
    (HUGE_VAL)
    To allow all numeric types to match, use -relaxtypes.
calculate.c:46:4: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:47:10: Return value type double does not match declared type float:
    (pow(Numeral, SecondNumeral))
calculate.c:50:9: Return value type double does not match declared type float:
    (sqrt(Numeral))
calculate.c:52:9: Return value type double does not match declared type float:
    (sin(Numeral))
calculate.c:54:9: Return value type double does not match declared type float:
    (cos(Numeral))
calculate.c:56:9: Return value type double does not match declared type float:
    (tan(Numeral))
calculate.c:60:10: Return value type double does not match declared type float:
    (HUGE_VAL)

Finished checking --- 15 code warnings
[narogozhina@narogozhina lab_prog]$
```

Рис. 15: splint calculate.c

```
Finished checking --- 15 code warnings
[narogozhina@narogozhina lab_prog]$ splint main.c
Splint 3.1.2 --- 23 Jul 2022

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:14:2: 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:16:13: 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:16:16: Corresponding format code
main.c:16:2: Return value (type int) ignored: scanf("%s", &Ope...

Finished checking --- 4 code warnings
[narogozhina@narogozhina lab_prog]$
```

Рис. 16: splint main.c

Выводы

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