

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

ОЗЪЯС Стив Икнэль Дани

НКНбд-02-21

3 June, 2022 Moscow, Russia

¹RUDN University, Moscow, Russian Federation

Цель работы

Цель данной 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.

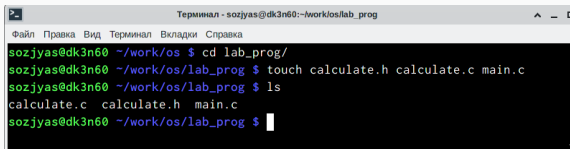
Ход работы

1. В домашнем каталоге создал подкаталог `~/work/os/lab_prog`. (рис. 1)

```
sozjyas@dk3n60 ~ $ cd work/  
sozjyas@dk3n60 ~/work $ ls  
os  
sozjyas@dk3n60 ~/work $ cd os  
sozjyas@dk3n60 ~/work/os $ mkdir lab_prog  
sozjyas@dk3n60 ~/work/os $ ls  
lab06 lab_prog
```

Figure 1: Создание подкаталога `~/work/os/lab_prog`

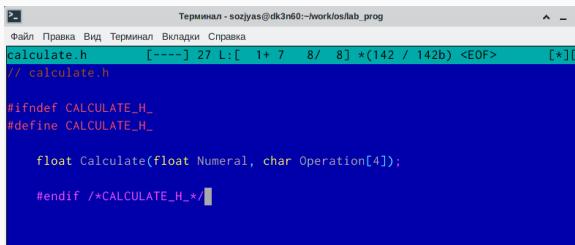
2. Создал в нём файлы: calculate.h, calculate.c, main.c. (рис. 2)



```
Терминал - sozjyas@dk3n60:~/work/os/lab_prog
Файл Правка Вид Терминал Вкладки Справка
sozjyas@dk3n60 ~/work/os $ cd lab_prog/
sozjyas@dk3n60 ~/work/os/lab_prog $ touch calculate.h calculate.c main.c
sozjyas@dk3n60 ~/work/os/lab_prog $ ls
calculate.c calculate.h main.c
sozjyas@dk3n60 ~/work/os/lab_prog $
```

Figure 2: Создание файлов calculate.h, calculate.c, main.c

- calculate.h, (рис. 3)



The image shows a terminal window with a blue background. The title bar at the top reads "Терминал - sozjyas@dk3n60:~/work/os/lab_prog". Below the title bar is a menu bar with the options "Файл", "Правка", "Вид", "Терминал", "Вкладки", and "Справка". The main area of the terminal displays the content of a file named "calculate.h". The first line is a comment: "// calculate.h". The second line is a preprocessor directive: "#ifndef CALCULATE_H_". The third line is another preprocessor directive: "#define CALCULATE_H_". The fourth line is a function declaration: "float Calculate(float Numeral, char Operation[4]);". The fifth line is a preprocessor directive: "#endif /*CALCULATE_H_*/". The cursor is positioned at the end of the fifth line.

```
calculate.h [----] 27 L: [ 1+ 7 8/ 8] *(142 / 142b) <EOF> [*]  
// calculate.h  
  
#ifndef CALCULATE_H_  
#define CALCULATE_H_  
  
    float Calculate(float Numeral, char Operation[4]);  
  
#endif /*CALCULATE_H_*/
```

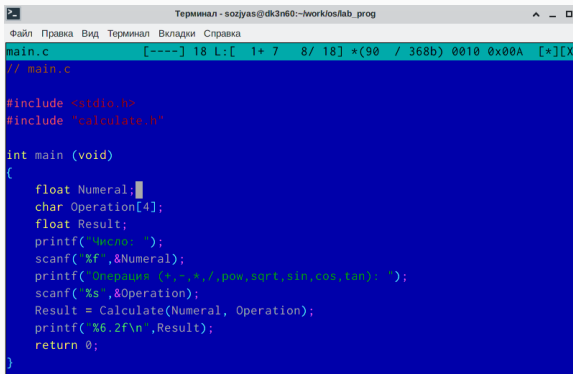
Figure 3: calculate.h

- calculate.c, (рис. 4)

Figure 4: calculate.c

6/29

- main.c (рис. 5)



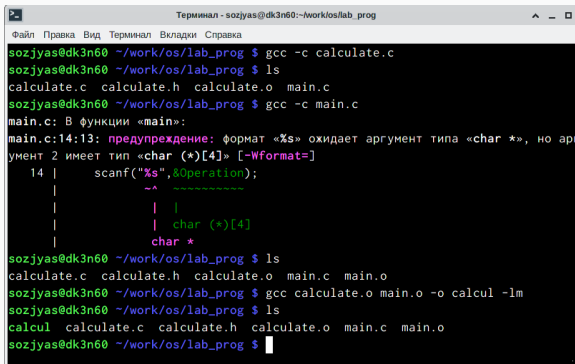
```
Терминал - sozjyas@dk3n60:~/work/os/lab_prog
Файл  Правка  Вид  Терминал  Вкладки  Справка
main.c [----] 18 L: [ 1+ 7 8/ 18] *(90 / 368b) 0010 0x00A [*][X]
// main.c

#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 5: main.c

3. Выполнил компиляцию программы посредством gcc (рис. 6)



```
Терминал - sozjyas@dk3n60:~/work/os/lab_prog
Файл  Правка  Вид  Терминал  Вкладки  Справка
sozjyas@dk3n60 ~/work/os/lab_prog $ gcc -c calculate.c
sozjyas@dk3n60 ~/work/os/lab_prog $ ls
calculate.c calculate.h calculate.o main.c
sozjyas@dk3n60 ~/work/os/lab_prog $ gcc -c main.c
main.c: В функции «main»:
main.c:14:13: предупреждение: формат «%s» ожидает аргумент типа «char *», но аргумент 2 имеет тип «char (*)[4]» [-Wformat=]
   14 |     scanf("%s",&operation);
      |           ^~ ~~~~~
      |           | |
      |           | | char (*)[4]
      |           | |
      |           | char *
sozjyas@dk3n60 ~/work/os/lab_prog $ ls
calculate.c calculate.h calculate.o main.c main.o
sozjyas@dk3n60 ~/work/os/lab_prog $ gcc calculate.o main.o -o calcul -lm
sozjyas@dk3n60 ~/work/os/lab_prog $ ls
calcul calculate.c calculate.h calculate.o main.c main.o
sozjyas@dk3n60 ~/work/os/lab_prog $
```

Figure 6: Выполнение компиляции программы посредством gcc

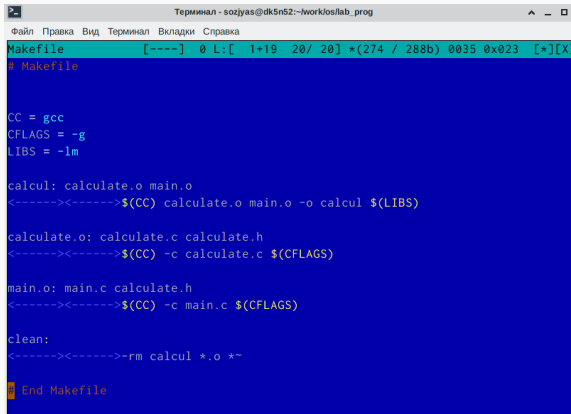
4. Я исправил незначительную синтаксическую ошибку.

5. Создал Makefile с заданным содержанием

- CC = gcc замена слова gcc на CC
- LIBS = -lm дополнительные опции
- calcul: calculate.o main.o
 - gcc calculate.o main.o -o calcul \$(LIBS)
 - Это команда для создания исполняемого файла calcul

- calculate.o: calculate.c calculate.h
 - gcc -c calculate.c \$(CFLAGS)
 - Это команда для создания объектного файла calculate.o (рис. 7)
- main.o: main.c calculate.h
 - gcc -c main.c \$(CFLAGS)
 - Это команда для создания объектного файла main.o

- clean:
 - -rm calcul.o ~
 - Это команда для удаления всех объектных файлов и файлов с знаком ~ в конце



```
Терминал - sozjyas@dk5n52:~/work/ios/lab_prog
Файл Правка Вид Терминал Вкладки Справка
Makefile [----] 0 L: [ 1+19 20/ 20] *(274 / 288b) 0035 0x023 [*][X]
# Makefile

CC = gcc
CFLAGS = -g
LIBS = -lm

calcul: calculate.o main.o
<-----><----->$(CC) calculate.o main.o -o calcul $(LIBS)

calculate.o: calculate.c calculate.h
<-----><----->$(CC) -c calculate.c $(CFLAGS)

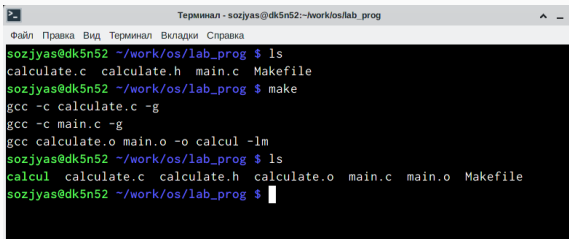
main.o: main.c calculate.h
<-----><----->$(CC) -c main.c $(CFLAGS)

clean:
<-----><----->-rm calcul *.o *~

End Makefile
```

Figure 7: Создание Makefile

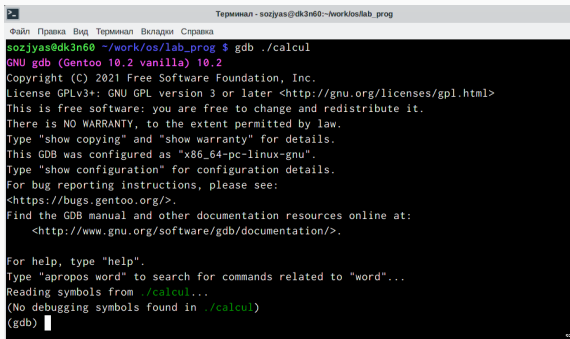
6. С помощью gdb выполнил отладку программы calcul (перед использованием gdb исправил Makefile)(рис. 8



```
Терминал - sozjyas@dk5n52:~/work/os/lab_prog
Файл Правка Вид Терминал Вкладки Справка
sozjyas@dk5n52 ~/work/os/lab_prog $ ls
calculate.c calculate.h main.c Makefile
sozjyas@dk5n52 ~/work/os/lab_prog $ make
gcc -c calculate.c -g
gcc -c main.c -g
gcc calculate.o main.o -o calcul -lm
sozjyas@dk5n52 ~/work/os/lab_prog $ ls
calcul calculate.c calculate.h calculate.o main.c main.o Makefile
sozjyas@dk5n52 ~/work/os/lab_prog $
```

Figure 8: Выполнение отладки программы calcul

- Запустил отладчик GDB, загрузив в него программу для отладки:(рис. 9



```
Терминал - sozjyas@dk3n60:~/work/os/lab_prog
Файл  Правка  Вид  Терминал  Вкладки  Справка
sozjyas@dk3n60 ~/work/os/lab_prog $ gdb ./calcul
GNU gdb (Gentoo 10.2 vanilla) 10.2
Copyright (C) 2021 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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://bugs.gentoo.org/>.
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...
(No debugging symbols found in ./calcul)
(gdb) 
```

Figure 9: Запуск отладчика GDB

- Запустил программу внутри отладчика (рис. 10)

```
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./calcul...
(No debugging symbols found in ./calcul)
(gdb) run
Starting program: /afs/.dk.sci.pfu.edu.ru/home/s/o/sozjyas/work/os/lab_prog/calcul
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 6
11.00
[Inferior 1 (process 20865) exited normally]
```

Figure 10: Запуск программы внутри отладчика

- Просмотрел исходный код используя команду list (рис. 11)

```
(gdb) list
warning: Source file is more recent than executable.
1      // main.c
2
3      #include <stdio.h>
4      #include "calculate.h"
5
6      int main (void)
7      {
8          float Numeral;
9          char Operation[4];
10         float Result;
```

Figure 11: Просмотр исходного кода

- Просмотрел строки с 12 по 15 основного файла (рис. 12)

```
(gdb) list 12,15
12     scanf("%f",&Numeral);
13     printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
14     scanf("%s",Operation);
15     Result = Calculate(Numeral, Operation);
(gdb)
```

Figure 12: Просмотр строк с 12 по 15 основного файла

- Просмотрел определённые строки не основного файла (рис. 13)

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

Figure 13: Просмотр определённых строк не основного файла

- Установил точку останова в файле calculate.c на строке номер 21:
(рис. 14)

```
(gdb) b 21  
Breakpoint 1 at 0x1252: file calculate.c, line 21.  
(gdb) █
```

Figure 14: Установка точки останова в файле calculate.c

- Вывел информацию об имеющихся в проекте точка останова: (рис. 15)

```
(gdb) i b
Num      Type      Disp Enb Address          What
1        breakpoint keep y   0x0000000000001252 in Calculate at calculate.c:21
(gdb)
```

Figure 15: Вывод информации о точке останова

- Запустил программу внутри отладчика и убедился, что программа остановился в момент прохождения точки останова: (рис. 16)

```
(gdb) run
Starting program: /afs/.dk.sci.pfu.edu.ru/home/s/o/sozjyas/work/os/lab_prog/calcul
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -

Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffff0c4 "-") at calculate.c:21
21      scanf("%f",&SecondNumeral);
```

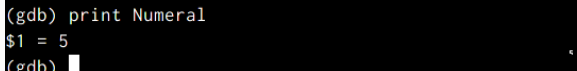
Figure 16: Запуск программы внутри отладчика при наличии точки останова

- backtrace (рис. 17)

```
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffff0c4 "-") at calculate.c:21
#1 0x0000555555555566 in main () at main.c:15
(gdb)
```

Figure 17: backtrace

- Посмотрел, чему равно на этом этапе значение переменной Numeral (рис. 18)



```
(gdb) print Numeral
$1 = 5
(gdb) 
```

Figure 18: Значение переменной Numeral на этом этапе

- Сравнил с результатом вывода на экран (рис. 19)

```
(gdb) display Numeral  
1: Numeral = 5  
(gdb) █
```

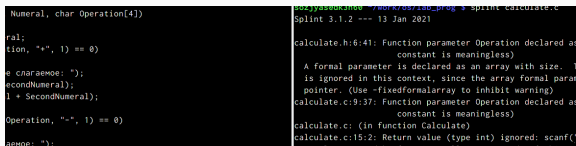
Figure 19: Сравнение с результатом вывода на экран после использования команды display

- Убрал точку останова: (рис. 20)

```
(gdb) delete 1  
(gdb) i b  
No breakpoints or watchpoints.
```

Figure 20: Удаление точки останова

7. С помощью утилиты splint попробуйте проанализировать коды файлов:
- calculate.c (рис. 21)



```
Numeral, char Operation[4])
{
    if (Operation[0] != '+' || Operation[1] != '-' || Operation[2] != '*' || Operation[3] != '/')
        return 0;
    if (Operation[0] == '+')
        SecondNumeral = SecondNumeral + FirstNumeral;
    else if (Operation[0] == '-')
        SecondNumeral = SecondNumeral - FirstNumeral;
    else if (Operation[0] == '*')
        SecondNumeral = SecondNumeral * FirstNumeral;
    else if (Operation[0] == '/')
        SecondNumeral = SecondNumeral / FirstNumeral;
    printf("Result: %d\n", SecondNumeral);
    return 1;
}
```

```
baz@yashukhino:~/0006/09/180_prog$ splint calculate.c
Splint 3.1.2 --- 13 Jan 2021

calculate.h:6:41: Function parameter Operation declared as
        constant is meaningless)
A formal parameter is declared as an array with size. The
size is ignored in this context, since the array formal parameter
is a pointer. (Use -fixedformalarray to inhibit warning)
calculate.c:9:37: Function parameter Operation declared as
        constant is meaningless)
calculate.c: (in function Calculate)
calculate.c:15:2: Return value (type int) ignored: scanf("%d", &FirstNumeral);
```

Figure 21: Анализ кода файла calculate.c

- и main.c. (рис. 22)

```
sozjyas@dk3n60 ~/work/os/lab_prog $ splint main.c
Splint 3.1.2 --- 13 Jan 2021

calculate.h:6:41: 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:12:5: 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:14:5: Return value (type int) ignored: scanf("%s", Oper...

Finished checking --- 3 code warnings
```

Figure 22: Анализ кода файла main.c

Выводы

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

Wer's nicht glaubt, bezahlt einen Taler