

Лабораторная работа №13

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

Медникова Е.М.

06 мая 2023

Российский университет дружбы народов, Москва, Россия

Факультет физико-математических и естественных наук

Информация

- Медникова Екатерина Михайловна
- студентка направления бакалавриата 01.03.00 Математика и механика
- Российский университет дружбы народов
- 1132226549@rudn.ru

Цели и задачи

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.

Выполнение лабораторной работы

В домашнем каталоге создала подкаталог ~/work/os/lab_prog.

```
[emmednikova@fedora ~]$ cd work/os  
[emmednikova@fedora os]$ mkdir lab_prog  
[emmednikova@fedora os]$ cd lab_prog/  
[emmednikova@fedora lab_prog]$
```

Создала в нём файлы: calculate.h, calculate.c, main.c.

```
[emmednikova@fedora lab_prog]$ touch calculate.h calculate.c main.c  
[emmednikova@fedora lab_prog]$ ls  
calculate.c calculate.h main.c  
[emmednikova@fedora lab_prog]$
```


Написала программы в созданные файлы.

```
Открыть ▾ + *calculate.c
~\work\os\lab_prog

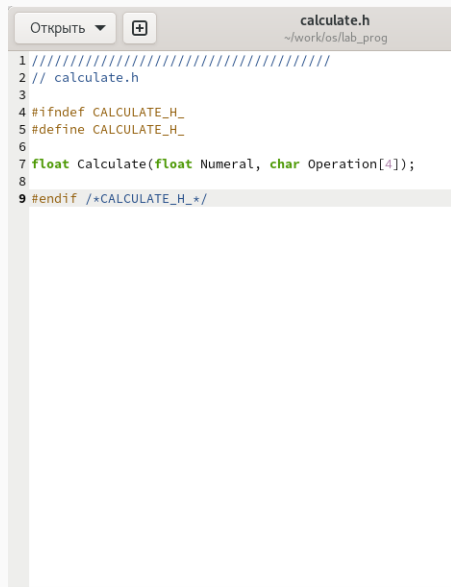
1 //////////////////////////////////////////////////
2 // calculate.c
3
4 #include <stdio.h>
5 #include <math.h>
6 #include <string.h>
7 #include "calculate.h"
8
9 float
10 Calculate(float Numeral, char Operation[4])
11 {
12     float SecondNumeral;
13     if(strncmp(Operation, "+", 1) == 0)
14     {
15         printf("Второе слагаемое: ");
16         scanf("%f",&SecondNumeral);
17         return(Numeral + SecondNumeral);
18     }
19     else if(strncmp(Operation, "-", 1) == 0)
20     {
21         printf("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);
30     }
31     else if(strncmp(Operation, "/", 1) == 0)
```

Написала программы в созданные файлы.

```
Открыть ▾ + *calculate.c
~/work/os/lab_prog

32 {
33     printf("Делитель: ");
34     scanf("%f",&SecondNumeral);
35     if(SecondNumeral == 0)
36     {
37         printf("Ошибка: деление на ноль! ");
38         return(HUGE_VAL);
39     }
40     else
41         return(Numeral / SecondNumeral);
42 }
43 else if(strncmp(Operation, "pow", 3) == 0)
44 {
45     printf("Степень: ");
46     scanf("%f",&SecondNumeral);
47     return(pow(Numeral, SecondNumeral));
48 }
49 else if(strncmp(Operation, "sqrt", 4) == 0)
50     return(sqrt(Numeral));
51 else if(strncmp(Operation, "sin", 3) == 0)
52     return(sin(Numeral));
53 else if(strncmp(Operation, "cos", 3) == 0)
54     return(cos(Numeral));
55 else if(strncmp(Operation, "tan", 3) == 0)
56     return(tan(Numeral));
57 else
58 {
59     printf("Неправильно введено действие ");
60     return(HUGE_VAL);
61 }
62 }
```

Написала программы в созданные файлы.



The screenshot shows a code editor window with a title bar that includes a dropdown menu labeled "Открыть" (Open) and a plus icon. The file name "calculate.h" is displayed in the title bar, along with the path "~/work/os/lab_prog". The code is written in C and includes a header guard. The visible code is as follows:

```
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_*/
```

Написала программы в созданные файлы.

```
main.c
~/work/os/lab_prog

1 ///////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
5 #include "calculate.h"
6
7 int main (void)
8 {
9     float Numeral;
10    char Operation[4];
11    float Result;
12    printf("Число: ");
13    scanf("%f",&Numeral);
14    printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
15    scanf("%s",&Operation);
16    Result = Calculate(Numeral, Operation);
17    printf("%6.2f\n",Result);
18    return 0;
19 }
```


Выполнила компиляцию программы посредством gcc.

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

При компиляции программы синтаксических ошибок выявлено не было.

```
[emmednikova@fedora lab_prog]$ touch Makefile
[emmednikova@fedora lab_prog]$ ls
calcul calculate.c calculate.h calculate.o main.c main.o Makefile
[emmednikova@fedora lab_prog]$
```

Написала программу в данный файл.


Открыть ▾ 

Makefile
~/work/os/lab_prog

```
1 #
2 # Makefile
3 #
4
5 CC = gcc
6 CFLAGS =
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
```


Данный файл нужен для автоматической компиляции файлов `calculate.c`, `main.c`, а также их объединения в один исполняемый файл `calcul`. Функция `clean` - автоматическое удаление файлов. Переменная `CC` отвечает за утилиту для компиляции. Переменная `CFLAGS` отвечает за опции в данной утилите. Переменная `LIBS` отвечает за опции для объединения объектных файлов в один исполняемый файл.

Перед использованием gdb исправила Makefile.

Открыть ▾ 

Makefile
~/work/os/lab_prog

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

Выполнила компиляцию файлов.

```
[emmednikova@fedora lab_prog]$ make clean
rm calcul *.o *~
rm: невозможно удалить '*~': Нет такого файла или каталога
make: [Makefile:19: clean] Ошибка 1 (игнорирование)
[emmednikova@fedora lab_prog]$ make calculate.o
gcc -c calculate.c -g
[emmednikova@fedora lab_prog]$ make main.o
gcc -c main.c -g
[emmednikova@fedora lab_prog]$ make calcul
gcc calculate.o main.o -o calcul -lm
```

С помощью gdb выполнила отладку программы calcul.

```
[emmednikova@fedora lab_prog]$ gdb ./calcul
GNU gdb (GDB) Fedora Linux 13.1-3.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)
```

Для запуска программы внутри отладчика ввела команду run.

```
(gdb) run
Starting program: /home/emmednikova/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.
Downloading separate debug info for system-supplied DSO at 0x7ffff7fc6000
Downloading separate debug info for /lib64/libm.so.6
Downloading separate debug info for /lib64/libc.so.6
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 3
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 8
  11.00
[Inferior 1 (process 3929) exited normally]
(gdb)
```

Для постраничного (по 9 строк) просмотра исходного код использовала команду list.

```
(gdb) list
1  ////////////////////////////////////////////
2  // main.c
3
4  #include <stdio.h>
5  #include "calculate.h"
6
7  int main (void)
8  {
9      float Numeral;
10     char Operation[4];
(gdb) list
11     float Result;
12     printf("Число: ");
13     scanf("%f",&Numeral);
14     printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
15     scanf("%s",&Operation);
16     Result = Calculate(Numeral, Operation);
17     printf("%6.2f\n",Result);
18     return 0;
19 }
(gdb)
```

Для просмотра строк с 12 по 15 основного файла использовала list с параметрами.

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

Для просмотра определённых строк не основного файла использовала list с параметрами.

```
(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) █
```


Установила точку останова в файле calculate.c на строке номер 21.

```
(gdb) break 21  
Breakpoint 1 at 0x40120f: file calculate.c, line 21.  
(gdb)
```

Вывела информацию об имеющихся в проекте точках останова.

```
(gdb) info breakpoints
```

Num	Type	Disp	Enb	Address	What
1	breakpoint	keep y		0x000000000040120f	in <code>Calculate</code> at <code>calculate.c:21</code>

```
(gdb) █
```

Запустила программу внутри отладчика и убедилась, что программа остановится в момент прохождения точки останова.

```
(gdb) run
Starting program: /home/emmednikova/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=0x7fffffffdf24 "-")
    at calculate.c:21
21      printf("Вычитаемое: ");
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffffdf24 "-") at calculate.c:21
#1 0x00000000004014eb in main () at main.c:16
(gdb)
```

Отладчик выдал следующую информацию:

```
(gdb) run
Starting program: /home/emmednikova/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=0x7fffffffdf24 "-")
    at calculate.c:21
21      printf("Вычитаемое: ");
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffffdf24 "-") at calculate.c:21
#1 0x00000000004014eb in main () at main.c:16
(gdb)
```

Посмотрела, чему равно на этом этапе значение переменной Numeral, введя команду `print Numeral`. На экран было выведено число 5.

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

После использования команды `display Numeral` на экран также было выведено число 5.

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

```
(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) 
```

С помощью утилиты splint было замечено, что в файлах calculate.c и main.c есть функция scanf, которая возвращает целое число, но данные числа не используются и нигде не сохраняются. Далее утилита вывела предупреждение о том, что в файле calculate.c происходит сравнение вещественного числа с нулём.

```
[emmednikova@fedora lab_prog]$ splint calculate.c
bash: splint: команда не найдена...
Установить пакет «splint», предоставляющий команду «splint»? [N/y] y

* Ожидание в очереди...
* Загрузка списка пакетов....
Следующие пакеты должны быть установлены:
 splint-3.1.2-29.fc37.x86_64    An implementation of the lint program
Продолжить с этими изменениями? [N/y] y

* Ожидание в очереди...
* Ожидание аутентификации...
* Ожидание в очереди...
* Загрузка пакетов...
* Запрос данных...
* Проверка изменений...
* Установка пакетов...
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
```


С помощью утилиты splint было замечено, что в файлах calculate.c и main.c есть функция scanf, которая возвращает целое число, но данные числа не используются и нигде не сохраняются. Далее утилита вывела предупреждение о том, что в файле calculate.c происходит сравнение вещественного числа с нулём.

```
calculate.c:10:31: Function parameter operation declared as manifest array
      (size constant is meaningless)
calculate.c: (in function Calculate)
calculate.c:16:7: 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:22:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:10: 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:38:17: Return value type double does not match declared type float:
      (HUGE_VAL)
      To allow all numeric types to match, use +relaxtypes.
calculate.c:46:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:47:13: Return value type double does not match declared type float:
      (pow(Numeral, SecondNumeral))
calculate.c:50:11: Return value type double does not match declared type float:
      (sqrt(Numeral))
calculate.c:52:11: Return value type double does not match declared type float:
      (sin(Numeral))
calculate.c:54:11: Return value type double does not match declared type float:
```

С помощью утилиты splint было замечено, что в файлах calculate.c и main.c есть функция scanf, которая возвращает целое число, но данные числа не используются и нигде не сохраняются. Далее утилита вывела предупреждение о том, что в файле calculate.c происходит сравнение вещественного числа с нулём.

```
[emmednikova@fedora 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:13:3: 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:15:14: 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)
```

Выводы

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