

# Средства для создания приложений в ОС UNIX.

---

Евгений Алексеевич Лисягин НБИбд-01-20<sup>1</sup>

3 июня, 2021, Москва, Россия

<sup>1</sup>Российский Университет Дружбы Народов

# Цели и задачи работы

---

## Цель лабораторной 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.

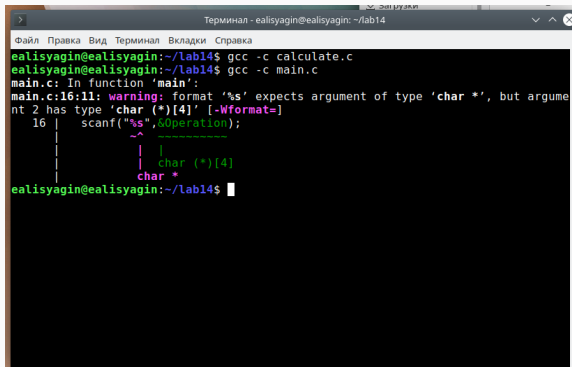
# Задачи лабораторной работы

- 1 Написать код приложения
- 2 Выполнить компиляцию
- 3 Подготовить Makefile
- 4 Выполнить отладку в GDB
- 5 Проанализировать код при помощи splint

# **Процесс выполнения лабораторной работы**

---

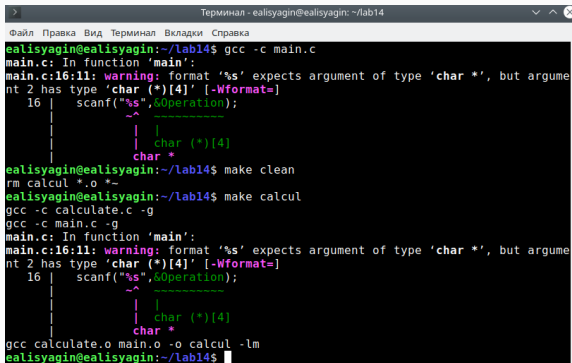
# Выполнение работы



```
Терминал - ealisyagin@ealisyagin: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
ealisyagin@ealisyagin:~/lab14$ gcc -c calculate.c
ealisyagin@ealisyagin:~/lab14$ gcc -c main.c
main.c: In function 'main':
main.c:16:11: warning: format '%s' expects argument of type 'char *', but argume
nt 2 has type 'char (*)[4]' [-Wformat=]
   16 |     scanf("%s",&operation);
      |           ^~
      |           | |
      |           | | char (*)[4]
      |           | | char *
ealisyagin@ealisyagin:~/lab14$
```

Figure 1: Компиляция

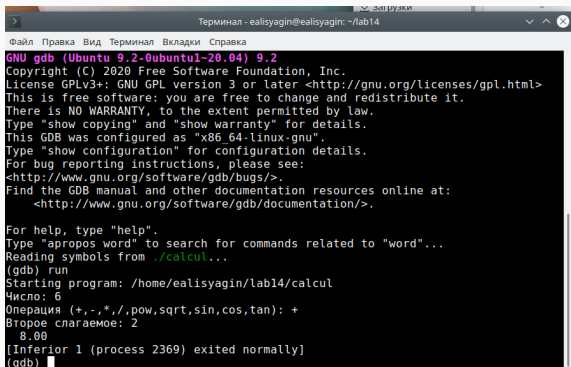
# Выполнение работы



```
Терминал - ealisyagin@ealisyagin: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
ealisyagin@ealisyagin:~/lab14$ gcc -c main.c
main.c: In function 'main':
main.c:16:11: warning: format '%s' expects argument of type 'char *', but argume
nt 2 has type 'char (*)[4]' [-Wformat=]
   16 |     scanf("%s",&operation);
      |           ^~
      |           |
      |           | char (*)[4]
      |           char *
ealisyagin@ealisyagin:~/lab14$ make clean
rm calcul *.o *-
ealisyagin@ealisyagin:~/lab14$ make calcul
gcc -c calculate.c -g
gcc -c main.c -g
main.c: In function 'main':
main.c:16:11: warning: format '%s' expects argument of type 'char *', but argume
nt 2 has type 'char (*)[4]' [-Wformat=]
   16 |     scanf("%s",&operation);
      |           ^~
      |           |
      |           | char (*)[4]
      |           char *
gcc calculate.o main.o -o calcul -lm
ealisyagin@ealisyagin:~/lab14$
```

Figure 2: Использование make

# Выполнение работы



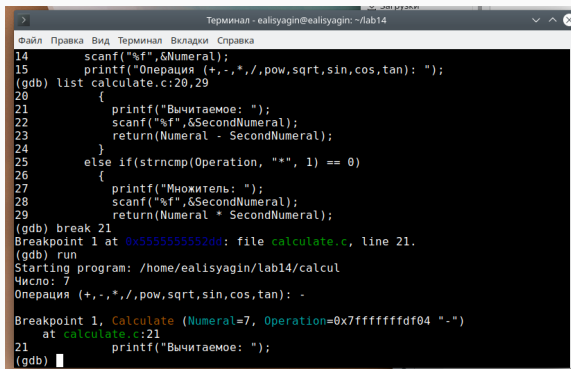
```
Терминал - ealisyagin@ealisyagin: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change 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-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://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/ealisyagin/lab14/calcul
Число: 6
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 2
      8.00
[Inferior 1 (process 2369) exited normally]
(gdb)
```

Figure 3: Использование отладчика



# Выполнение работы

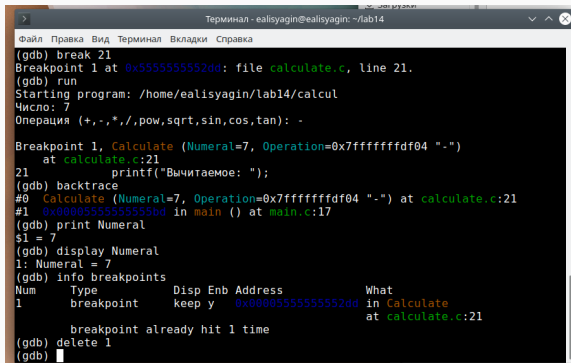


```
Терминал - ealisyagin@ealisyagin: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
14      scanf("%f",&Numeral);
15      printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
(gdb) list calculate.c:20,29
20      {
21          printf("Вычитаемое: ");
22          scanf("%f",&SecondNumeral);
23          return(Numeral - SecondNumeral);
24      }
25      else if(strncmp(Operation, "*", 1) == 0)
26      {
27          printf("Множитель: ");
28          scanf("%f",&SecondNumeral);
29          return(Numeral * SecondNumeral);
(gdb) break 21
Breakpoint 1 at 0x5555555552dd: file calculate.c, line 21.
(gdb) run
Starting program: /home/ealisyagin/lab14/calcul
Число: 7
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -

Breakpoint 1, Calculate (Numeral=7, Operation=0x7fffffffdf04 "-")
at calculate.c:21
21      printf("Вычитаемое: ");
(gdb) █
```

Figure 4: Использование отладчика

# Выполнение работы

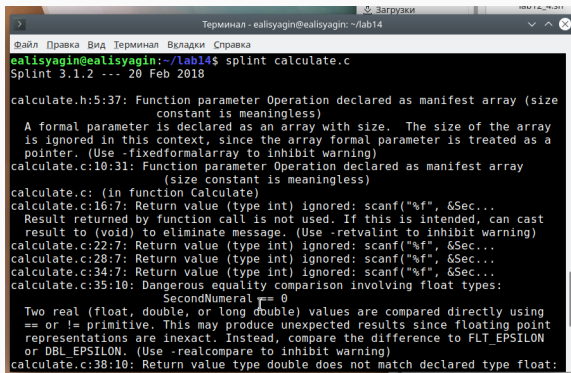
A screenshot of a terminal window titled "Терминал - ealisyagin@ealisyagin: ~/lab14". The window shows a GDB session for a program named "calcul". The user sets a breakpoint at line 21 of "calculate.c" and runs the program. The program outputs "Число: 7" and "Операция (+,-,\*,/,pow,sqrt,sin,cos,tan): -". The breakpoint is hit, and the user uses "backtrace" to see the call stack, "print Numeral" to see the value 7, "display Numeral" to watch the variable, and "info breakpoints" to see the current breakpoint status. Finally, the user deletes the breakpoint.

```
Терминал - ealisyagin@ealisyagin: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
(gdb) break 21
Breakpoint 1 at 0x555555552dd: file calculate.c, line 21.
(gdb) run
Starting program: /home/ealisyagin/lab14/calcul
Число: 7
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -

Breakpoint 1, Calculate (Numeral=7, Operation=0x7fffffffdf04 "-")
  at calculate.c:21
21      printf("Выводимое: ");
(gdb) backtrace
#0 Calculate (Numeral=7, Operation=0x7fffffffdf04 "-") at calculate.c:21
#1 0x00005555555555bd in main () at main.c:17
(gdb) print Numeral
$1 = 7
(gdb) display Numeral
1: Numeral = 7
(gdb) info breakpoints
Num   Type             Disp Enb Address                  What
1     breakpoint       keep y   0x00005555555555bd    in Calculate
                                           at calculate.c:21
      breakpoint already hit 1 time
(gdb) delete 1
(gdb)
```

Figure 5: Использование отладчика

# Выполнение работы



```
Терминал - ealisyagin@ealisyagin: ~/lab14
ealisyagin@ealisyagin:~/lab14$ splint calculate.c
Splint 3.1.2 --- 20 Feb 2018

calculate.h:5:37: 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:16:7: 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:38:10: Return value type double does not match declared type float:
```

Figure 6: Использование splint

## **Выводы по проделанной работе**

---

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.