

---

## Front matter

lang: ru-RU title: Средства, применяемые при разработке программного обеспечения в ОС типа UNIX/Linux author: | Арсоева Залина НБИБд-01-21\inst{1}

institute: | \inst{1}Российский Университет Дружбы Народов

date: 11 декабря, 2022, Москва, Россия

## Formatting

mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono toc: false slide\_level: 2 theme: metropolis header-includes:

- \metroset{progressbar=frametitle,sectionpage=progressbar,numbering=fraction}
- 'makeatletter'
- 'beamer@ignorenonframefalse'
- 'makeatother' aspectratio: 43 section-titles: true

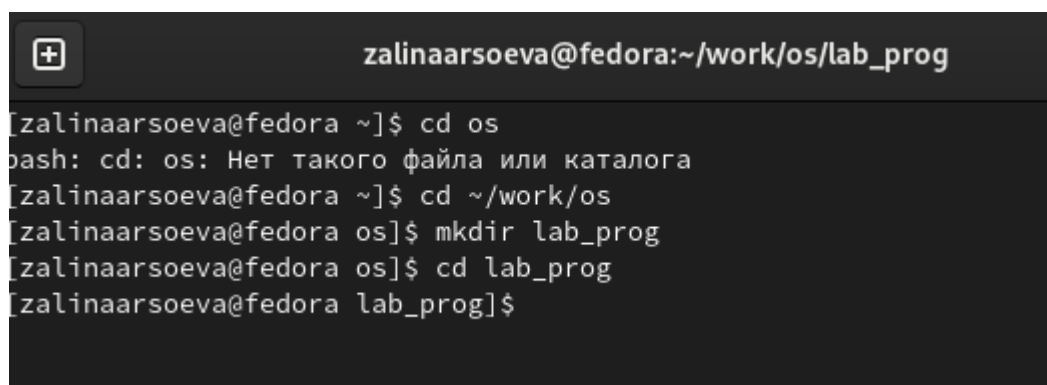
---

## Цель работы

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.

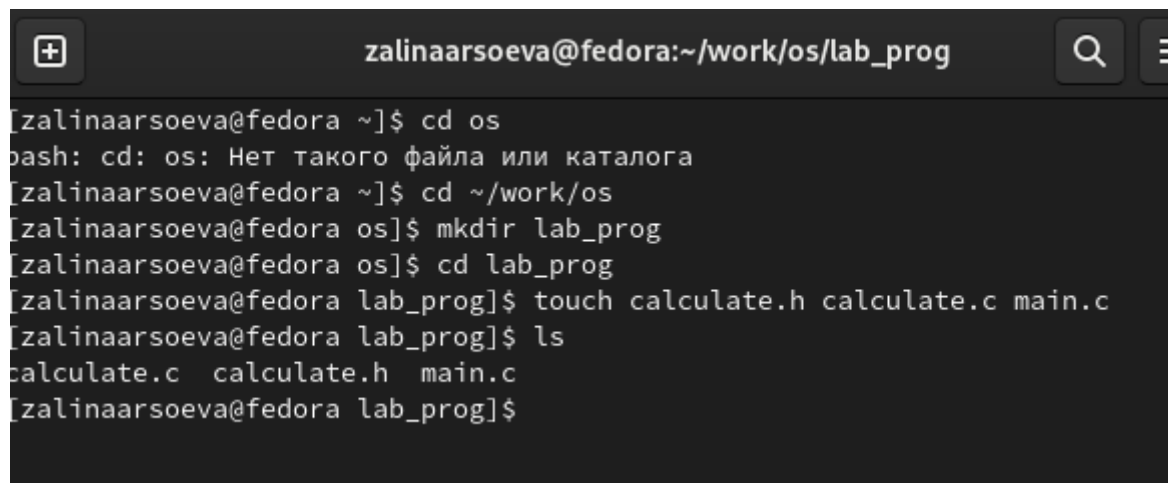
## Выполнение лабораторной работы

1. В домашнем каталоге создала подкаталог ~/work/os/lab\_prog.

A terminal window with a dark background. The title bar shows a window icon and the text 'zalinaarsoeva@fedora:~/work/os/lab\_prog'. The terminal content shows the following commands and output:

```
[zalinaarsoeva@fedora ~]$ cd os
bash: cd: os: Нет такого файла или каталога
[zalinaarsoeva@fedora ~]$ cd ~/work/os
[zalinaarsoeva@fedora os]$ mkdir lab_prog
[zalinaarsoeva@fedora os]$ cd lab_prog
[zalinaarsoeva@fedora lab_prog]$
```

2. Создала в нём файлы: calculate.h, calculate.c, main.c. Это примитивнейший калькулятор, способный складывать, вычитать, умножать, делить, возводить число в степень, вычислять квадратный корень, вычислять sin, cos, tan. При запуске он запрашивает первое число, операцию, второе число. После этого программа выводит результат и останавливается.

A terminal window with a dark background. The title bar shows the user 'zalinaarsoeva@fedora' and the current directory '~/work/os/lab\_prog'. The terminal content shows a series of commands and their outputs: 'cd os' results in an error message in Russian; 'cd ~/work/os' changes the directory; 'mkdir lab\_prog' creates the directory; 'cd lab\_prog' changes to the new directory; 'touch calculate.h calculate.c main.c' creates three files; 'ls' lists the files; and the prompt returns to the user in the 'lab\_prog' directory.

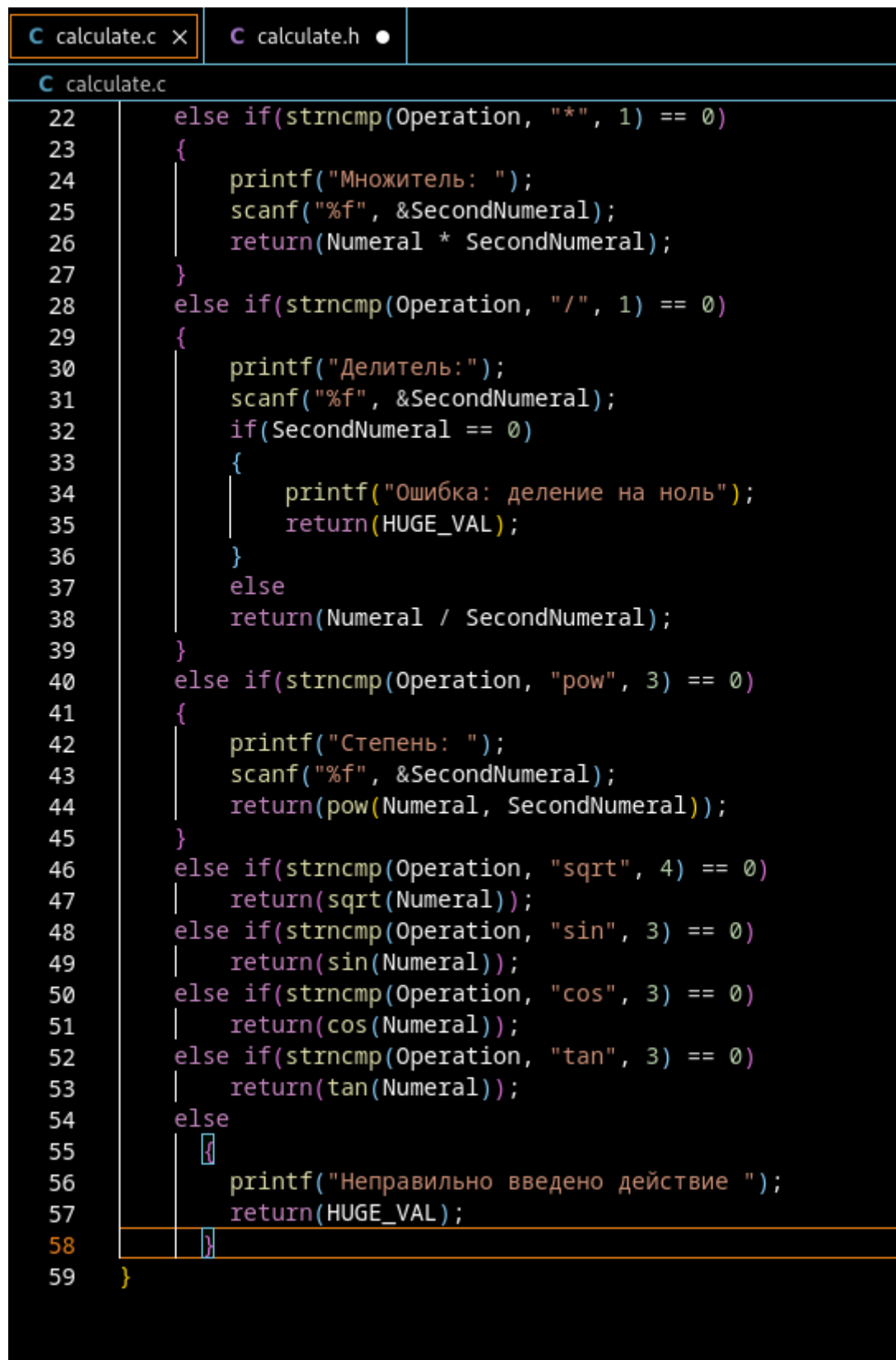
```
[zalinaarsoeva@fedora ~]$ cd os
bash: cd: os: Нет такого файла или каталога
[zalinaarsoeva@fedora ~]$ cd ~/work/os
[zalinaarsoeva@fedora os]$ mkdir lab_prog
[zalinaarsoeva@fedora os]$ cd lab_prog
[zalinaarsoeva@fedora lab_prog]$ touch calculate.h calculate.c main.c
[zalinaarsoeva@fedora lab_prog]$ ls
calculate.c calculate.h main.c
[zalinaarsoeva@fedora lab_prog]$
```

Реализация функций калькулятора в файле calculate.c:



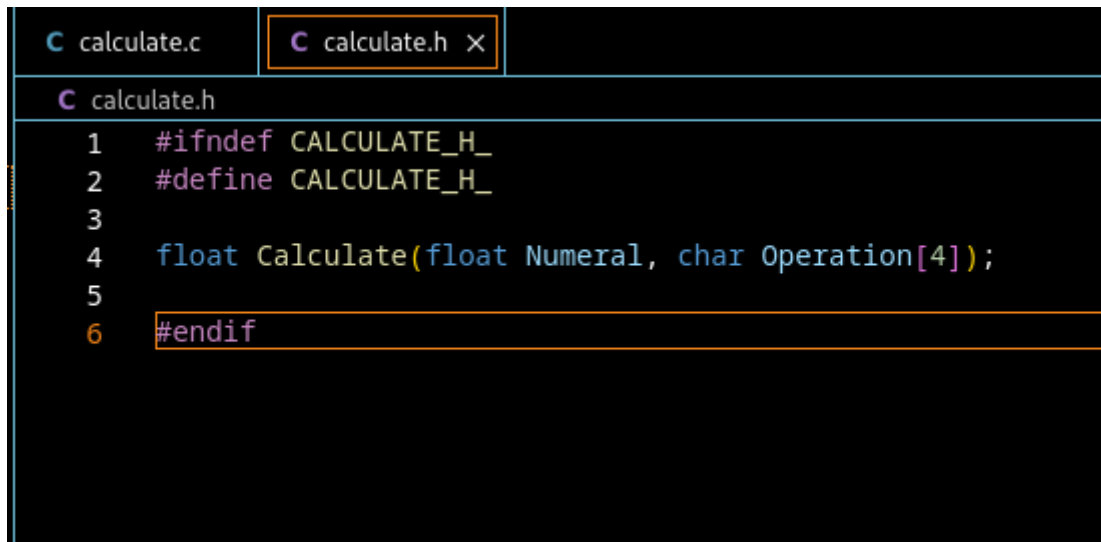
The image shows a code editor with two tabs: 'calculate.c' (active) and 'calculate.h'. The code in 'calculate.c' defines a function 'Calculate' that takes a float 'Numeral' and a character array 'Operation' of size 4. It implements basic arithmetic operations: addition (+), subtraction (-), multiplication (\*), and division (/). For division, it checks for a zero divisor and returns 'HUGE\_VAL' if it occurs. The code uses 'strncmp' for string comparison and 'scanf' to read the second operand. Comments in Russian are used to prompt the user for the second operand.

```
1  #include <stdio.h>
2  #include <math.h>
3  #include <string.h>
4  #include "calculate.h"
5
6  float
7  Calculate(float Numeral, char Operation[4])
8  {
9      float SecondNumeral;
10     if(strncmp(Operation, "+", 1) == 0)
11     {
12         printf("Второе слагаемое: ");
13         scanf("%f", &SecondNumeral);
14         return(Numeral + SecondNumeral);
15     }
16     else if(strncmp(Operation, "-", 1) == 0)
17     {
18         printf("Вычитаемое: ");
19         scanf("%f", &SecondNumeral);
20         return(Numeral - SecondNumeral);
21     }
22     else if(strncmp(Operation, "*", 1) == 0)
23     {
24         printf("Множитель: ");
25         scanf("%f", &SecondNumeral);
26         return(Numeral * SecondNumeral);
27     }
28     else if(strncmp(Operation, "/", 1) == 0)
29     {
30         printf("Делитель:");
31         scanf("%f", &SecondNumeral);
32         if(SecondNumeral == 0)
33         {
34             printf("Ошибка: деление на ноль");
35             return(HUGE_VAL);
36         }
37         else
38             return(Numeral / SecondNumeral);
39     }
40     else if(strncmp(Operation, "pow", 3) == 0)
```



```
calculate.c x calculate.h ●
C calculate.c
22     else if(strncmp(Operation, "*", 1) == 0)
23     {
24         printf("Множитель: ");
25         scanf("%f", &SecondNumeral);
26         return(Numeral * SecondNumeral);
27     }
28     else if(strncmp(Operation, "/", 1) == 0)
29     {
30         printf("Делитель:");
31         scanf("%f", &SecondNumeral);
32         if(SecondNumeral == 0)
33         {
34             printf("Ошибка: деление на ноль");
35             return(HUGE_VAL);
36         }
37         else
38             return(Numeral / SecondNumeral);
39     }
40     else if(strncmp(Operation, "pow", 3) == 0)
41     {
42         printf("Степень: ");
43         scanf("%f", &SecondNumeral);
44         return(pow(Numeral, SecondNumeral));
45     }
46     else if(strncmp(Operation, "sqrt", 4) == 0)
47         return(sqrt(Numeral));
48     else if(strncmp(Operation, "sin", 3) == 0)
49         return(sin(Numeral));
50     else if(strncmp(Operation, "cos", 3) == 0)
51         return(cos(Numeral));
52     else if(strncmp(Operation, "tan", 3) == 0)
53         return(tan(Numeral));
54     else
55     {
56         printf("Неправильно введено действие ");
57         return(HUGE_VAL);
58     }
59 }
```

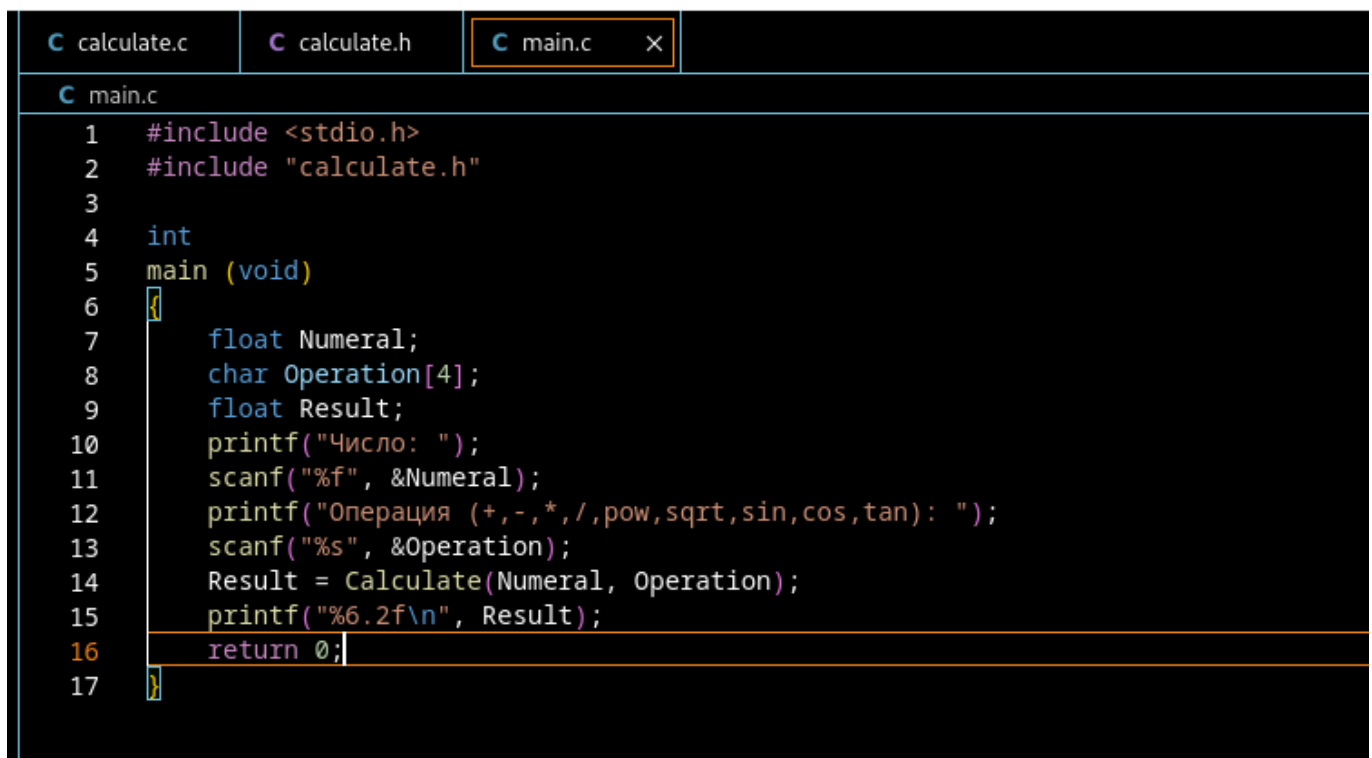
Интерфейсный файл calculate.h, описывающий формат вызова функции калькулятора:



The screenshot shows a code editor with three tabs at the top: 'calculate.c', 'calculate.h' (which is selected and highlighted with a yellow border), and an 'X' icon. The 'calculate.h' file contains the following code:

```
1  #ifndef CALCULATE_H_
2  #define CALCULATE_H_
3
4  float Calculate(float Numeral, char Operation[4]);
5
6  #endif
```

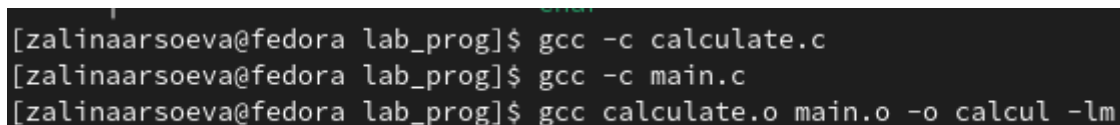
Основной файл main.c, реализующий интерфейс пользователя к калькулятору:



The screenshot shows a code editor with three tabs at the top: 'calculate.c', 'calculate.h', and 'main.c' (which is selected and highlighted with a yellow border). The 'main.c' file contains the following code:

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

3. Выполнила компиляцию программы посредством gcc:



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



calcul



calculate.c



calculate.h



calculate.o



main.c



main.o

4. Исправила синтаксические ошибки.

5. Создала Makefile

```
Открыть ▾ + Makefile
~/work/os/lab_prog

#
# 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
```

В содержании файла указаны флаги компиляции, тип компилятора и файлы, которые должен собрать сборщик.

6. С помощью gdb выполнила отладку программы calcul (перед использованием gdb исправил Makefile): – запустила отладчик GDB, загрузив в него программу для отладки: `gdb ./calcul` – для запуска программы внутри отладчика ввела команду `run`

```

*Minibuf-1* - GNU Emacs at fedora
File Edit Options Buffers Tools Minibuf Help
[Icons: File, Folder, Disk, Close, Save, Undo, Cut, Copy, Paste, Find]

#
# Makefile
#

CC = gcc
CFLAGS = -g
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

```

```

zalinaarsoeva@fedora:~/work/os/lab_prog — gdb ./calcul
Downloading separate debug info for /home/zalinaarsoeva/work/os/lab_prog/./calcul...
(No debugging symbols found in ./calcul)
(gdb) run
Starting program: /home/zalinaarsoeva/work/os/lab_prog/calcul
Downloading separate debug info for /home/zalinaarsoeva/work/os/lab_prog/system-supplied DSO at 0x7ffff7fc4000...
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: list
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): Неправильно введено действие   inf
[Inferior 1 (process 18116) exited normally]
(gdb) run
Starting program: /home/zalinaarsoeva/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): +
Второе слагаемое: 7
12.00
[Inferior 1 (process 18259) exited normally]
(gdb)

```

– для постраничного (по 9 строк) просмотра исходного код использовала команду list – для просмотра строк с 12 по 15 основного файла использовала list с параметрами: list 12,15

```
(gdb) list
1      #include <stdio.h>
2      #include "calculate.h"
3
4      int
5      main (void)
6      {
7          float Numeral;
8          char Operation[4];
9          float Result;
10         printf("Число: ");
(gdb)
```

```
(gdb) list 12,15
12         printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
13         scanf("%s",Operation);
14         Result = Calculate(Numeral,Operation);
15         printf("%6.2f\n",Result);
(gdb) list calculate.c:20,29
20         return(Numeral-SecondNumeral);
21     }
22     else if(strncmp(Operation,"*",1) == 0)
23     {
24         printf("Множитель: ");
25         scanf("%f",&SecondNumeral);
26         return(Numeral*SecondNumeral);
27     }
28     else if(strncmp(Operation,"/",1) == 0)
29     {
(gdb)
```

– для просмотра определённых строк не основного файла использовала list с параметрами: list calculate.c:20,29 – установила точку останова в файле calculate.c на строке номер 21: list calculate.c:20,27 break 20 – вывела информацию об имеющихся в проекте точка останова: info breakpoints



```
(gdb) list calculate.c:20,29
20         return(Numeral-SecondNumeral);
21     }
22     else if(strncmp(Operation,"*",1) == 0)
23     {
24         printf("Множитель: ");
25         scanf("%f",&SecondNumeral);
26         return(Numeral*SecondNumeral);
27     }
28     else if(strncmp(Operation,"/",1) == 0)
29     {
(gdb) list calculate.c:20,27
20         return(Numeral-SecondNumeral);
21     }
22     else if(strncmp(Operation,"*",1) == 0)
23     {
24         printf("Множитель: ");
25         scanf("%f",&SecondNumeral);
26         return(Numeral*SecondNumeral);
27     }
(gdb) break 21
Breakpoint 1 at 0x401247: file calculate.c, line 22.
```

```
(gdb) break 21
Note: breakpoint 1 also set at pc 0x401247.
Breakpoint 2 at 0x401247: file calculate.c, line 22.
(gdb) break 20
Breakpoint 3 at 0x401234: file calculate.c, line 20.
(gdb) break 21
Note: breakpoints 1 and 2 also set at pc 0x401247.
Breakpoint 4 at 0x401247: file calculate.c, line 22.
(gdb) print Numeral
No symbol "Numeral" in current context.
(gdb) run
Starting program: /home/abrovkin/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: backtrace

Breakpoint 3, Calculate (Numeral=5, Operation=0x7fffffffdee4 "-") at calculate.c:20
20         return(Numeral-SecondNumeral);
(gdb) print Numeral
$1 = 5
(gdb) display Numeral
1: Numeral = 5
(gdb) info breakpoints
Undefined info command: "braskpoints". Try "help info".
(gdb) info breakpoints
Num  Type             Disp Enb Address                                     What
1     breakpoint        keep y   0x0000000000401247 in Calculate at calculate.c:22
2     breakpoint        keep y   0x0000000000401247 in Calculate at calculate.c:22
3     breakpoint        keep y   0x0000000000401234 in Calculate at calculate.c:20
      breakpoint already hit 1 time
4     breakpoint        keep y   0x0000000000401247 in Calculate at calculate.c:22
(gdb) delete 1
(gdb) delete 2
(gdb) delete 3
```

– запустила программу внутри отладчика и убедилась, что программа остановится в момент прохождения точки останова – отладчик выдал следующую информацию, а команда backtrace показала весь стек вызываемых функций от начала программы до текущего места: – посмотрела, чему равно на этом этапе значение переменной Numeral, введя: print Numeral – сравнила с результатом вывода на экран после использования команды: display Numeral – убрала точки останова: info breakpoints delete 1

```
(gdb) run
Starting program: /home/abrovkin/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: backtrace

Breakpoint 3, Calculate (Numeral=5, Operation=0x7fffffffdee4 "-") at calculate.c:20
20      return(Numeral-SecondNumeral);
(gdb)
```

```
(gdb) list calculate.c:20,27
20      return(Numeral-SecondNumeral);
21  }
22  else if(strcmp(Operation,"*",1) == 0)
23  {
24      printf("Множитель: ");
25      scanf("%f",&SecondNumeral);
26      return(Numeral*SecondNumeral);
27  }
(gdb) break 21
Note: breakpoint 1 also set at pc 0x401247.
Breakpoint 2 at 0x401247: file calculate.c, line 22.
(gdb) break 20
Breakpoint 3 at 0x401234: file calculate.c, line 20.
(gdb) break 21
Note: breakpoints 1 and 2 also set at pc 0x401247.
Breakpoint 4 at 0x401247: file calculate.c, line 22.
(gdb) print Numeral
No symbol "Numeral" in current context.
(gdb) run
Starting program: /home/abrovkin/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: backtrace

Breakpoint 3, Calculate (Numeral=5, Operation=0x7fffffffdee4 "-") at calculate.c:20
20      return(Numeral-SecondNumeral);
(gdb) print Numeral
$1 = 5
(gdb)
```

```
(gdb) display Numeral
1: Numeral = 5
(gdb) info breakpoints
Undefined info command: "braskpoints". Try "help info".
(gdb) info breakpoints
Num  Type             Disp Enb Address                What
1    breakpoint        keep y  0x0000000000401247 in Calculate at calculate.c:22
2    breakpoint        keep y  0x0000000000401247 in Calculate at calculate.c:22
3    breakpoint        keep y  0x0000000000401234 in Calculate at calculate.c:20
    breakpoint already hit 1 time
4    breakpoint        keep y  0x0000000000401247 in Calculate at calculate.c:22
(gdb) delete 1
(gdb) delete 2
(gdb) delete 3
(gdb) delete 4
(gdb)
```

7. С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c.

```
[zalinaarsoeva@fedora lab_prog]$ splint calculate.c
Splint 3.1.2 --- 22 Jan 2022

calculate.h:4:37: 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:7:31: Function parameter Operation declared as manifest array (size
      constant is meaningless)
calculate.c: (in function Calculate)
calculate.c:13:9: 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:19:9: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:21:17: Parse Error. (For help on parse errors, see splint -help
      parseerrors.)
*** Cannot continue.
[zalinaarsoeva@fedora lab_prog]$
```

```
[zalinaarsoeva@fedora lab_prog]$ splint main.c
Splint 3.1.2 --- 22 Jan 2022

calculate.h:4:37: 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:11:5: 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:13:17: 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:13:13: Corresponding format code
main.c:13:5: Return value (type int) ignored: scanf("%s", &Ope...

Finished checking --- 4 code warnings
[zalinaarsoeva@fedora lab_prog]$ ~
```

## Вывод:

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.