

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

Бровкин Александр НБИбд-01-21¹

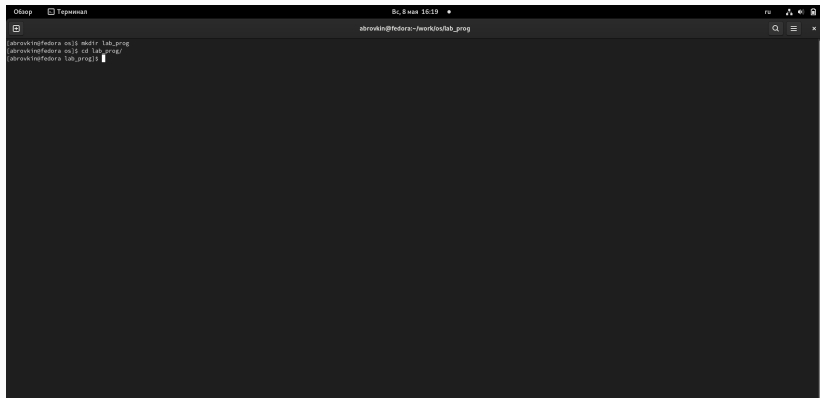
8 мая, 2022, Москва, Россия

¹Российский Университет Дружбы Народов

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

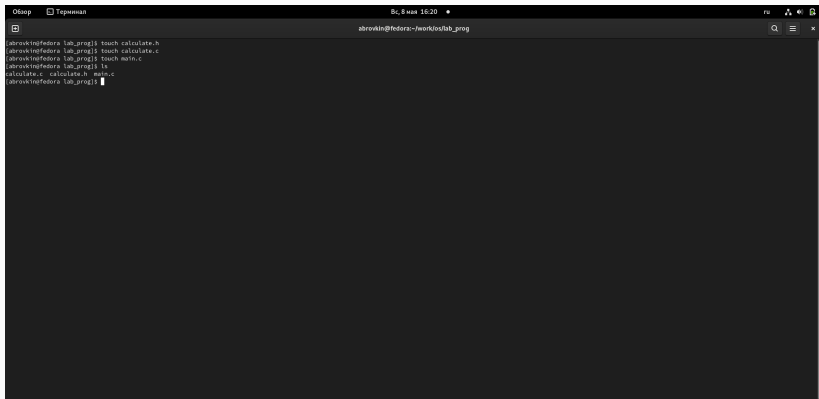
1. В домашнем каталоге создал подкаталог `~/work/os/lab_prog`.

A screenshot of a terminal window titled "Терминал" (Terminal). The window shows a user named "abrovkin@fedora" in the directory "~/.work/os/lab_prog". The user has executed the command "mkdir lab_prog" to create a new subdirectory. The terminal output shows the command being executed and the prompt returning to the user. The terminal window has a dark background and a light-colored text. The top bar of the window shows the title "Терминал", the user and host information "abrovkin@fedora: ~/.work/os/lab_prog", and the time "8:5 май 16:19". The bottom bar of the window shows the prompt "abrovkin@fedora lab_prog\$".

```
abrovkin@fedora ~$ mkdir lab_prog
abrovkin@fedora ~$ cd lab_prog/
abrovkin@fedora lab_prog$
```

Figure 1: Создаю подкаталог

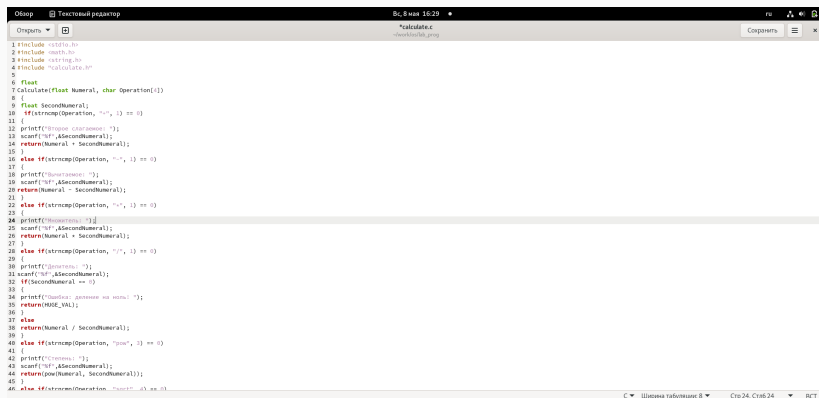
2. Создал в нём файлы: `calculate.h`, `calculate.c`, `main.c`. Это примитивнейший калькулятор, способный складывать, вычитать, умножать, делить, возводить число в степень, вычислять квадратный корень, вычислять \sin , \cos , \tan . При запуске он запрашивает первое число, операцию, второе число. После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

A terminal window titled "Терминал" (Terminal) with a dark background. The window shows a series of commands being entered at the prompt "abroavkin@fedora: ~/work/loss/lab_prog". The commands are: "touch calculate.h", "touch calculate.c", "touch math.c", "ls", and "calculate.c calculate.h main.c". The prompt is currently at the end of the last command.

```
abroavkin@fedora lab_prog$ touch calculate.h
abroavkin@fedora lab_prog$ touch calculate.c
abroavkin@fedora lab_prog$ touch math.c
abroavkin@fedora lab_prog$ ls
calculate.c calculate.h main.c
abroavkin@fedora lab_prog$
```

Figure 2: Создаю файлы

Реализация функций калькулятора в файле calculate.c

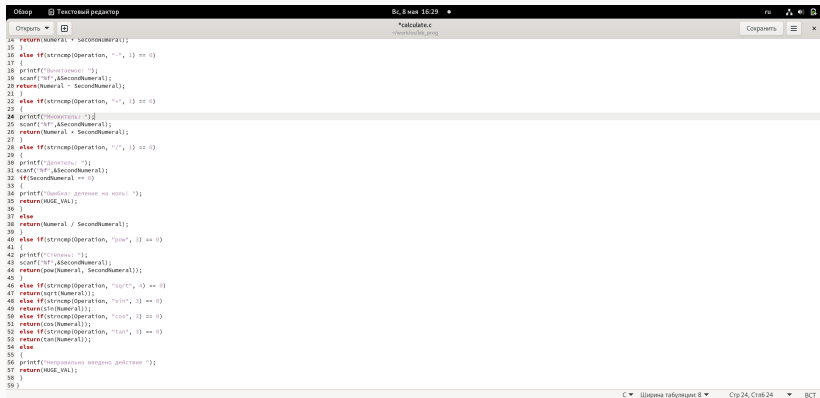


The screenshot shows a text editor window titled "Текстовый редактор" with a file named "calculate.c" open. The code implements a calculator with the following features:

- Includes: `<stdio.h>`, `<math.h>`, `<string.h>`, and `"calculate.h"`.
- Function: `Calculate(float Numeral, char Operation[])` returns a `float`.
- Operations: Addition (+), Subtraction (-), Multiplication (*), Division (/), Modulus (%), and Power (^).
- Flow: The function checks the operation character and performs the corresponding calculation using `scanf` to get the second number. It includes error handling for division by zero and modulus with non-positive numbers.

```
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[])
8 {
9     float SecondNumeral;
10    if(strlen(Operation) == 1)
11    {
12        printf("Введите второе число: ");
13        scanf("%f", &SecondNumeral);
14        return(Numeral + SecondNumeral);
15    }
16    else if(strlen(Operation) == 2)
17    {
18        printf("Введите второе число: ");
19        scanf("%f", &SecondNumeral);
20        return(Numeral - SecondNumeral);
21    }
22    else if(strlen(Operation) == 3)
23    {
24        printf("Введите второе число: ");
25        scanf("%f", &SecondNumeral);
26        return(Numeral * SecondNumeral);
27    }
28    else if(strlen(Operation) == 4)
29    {
30        printf("Введите второе число: ");
31        scanf("%f", &SecondNumeral);
32        if(SecondNumeral == 0)
33        {
34            printf("Деление на ноль!\n");
35            return(MAKE_VAL);
36        }
37        else
38            return(Numeral / SecondNumeral);
39    }
40    else if(strlen(Operation) == 5)
41    {
42        printf("Введите второе число: ");
43        scanf("%f", &SecondNumeral);
44        return(pow(Numeral, SecondNumeral));
45    }
46    else if(strlen(Operation) == 6)
47    {
48        printf("Введите второе число: ");
49        scanf("%f", &SecondNumeral);
50        return(fmod(Numeral, SecondNumeral));
51    }
```

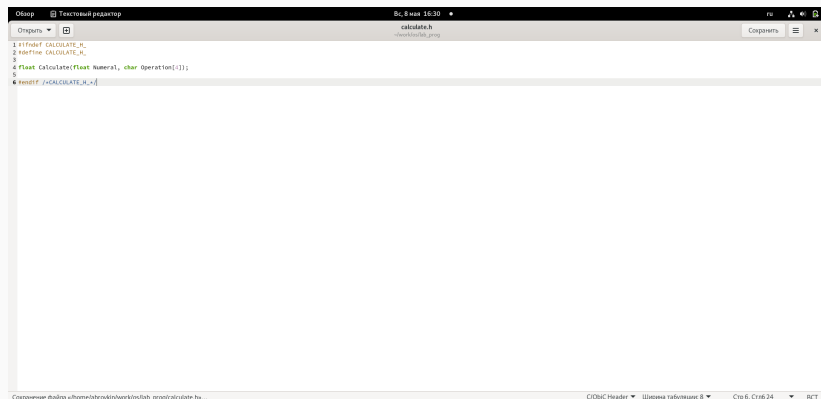
Figure 3: Скрипт



```
14 return (numeral + secondNumeral);
15 }
16 else if (strcmp(operation, "-") == 0)
17 {
18     printf("Вычитание: ");
19     scanf("%f", &secondNumeral);
20     return (numeral - secondNumeral);
21 }
22 else if (strcmp(operation, "*") == 0)
23 {
24     printf("Умножение: ");
25     scanf("%f", &secondNumeral);
26     return (numeral * secondNumeral);
27 }
28 else if (strcmp(operation, "/") == 0)
29 {
30     printf("Деление: ");
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 (strcmp(operation, "pow", 3) == 0)
41 {
42     printf("Степень: ");
43     scanf("%f", &secondNumeral);
44     return (pow(numeral, secondNumeral));
45 }
46 else if (strcmp(operation, "sqrt", 4) == 0)
47     return (sqrt(numeral));
48 else if (strcmp(operation, "sin", 3) == 0)
49     return (sin(numeral));
50 else if (strcmp(operation, "cos", 3) == 0)
51     return (cos(numeral));
52 else if (strcmp(operation, "tan", 3) == 0)
53     return (tan(numeral));
54 else
55 {
56     printf("Неправильно введено действие");
57     return (HUGE_VAL);
58 }
59 }
```

Figure 4: Пишу скрипт

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



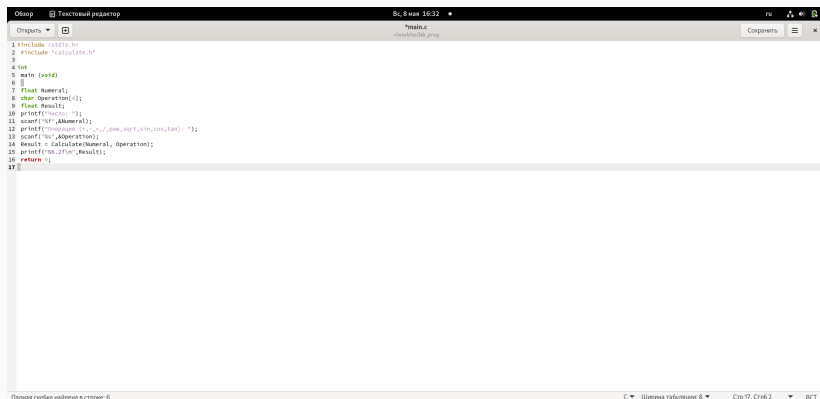
The image shows a screenshot of a text editor window titled "calculate.h". The editor displays the following C code:

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

The status bar at the bottom indicates the file path: "Сохранение файла «home\abrovkin\work\lab_prog\calculate.h»...", the CIOBJS Header, a tab width of 8, and the current position: "Стр 6, Стр 24".

Figure 5: Скрипт

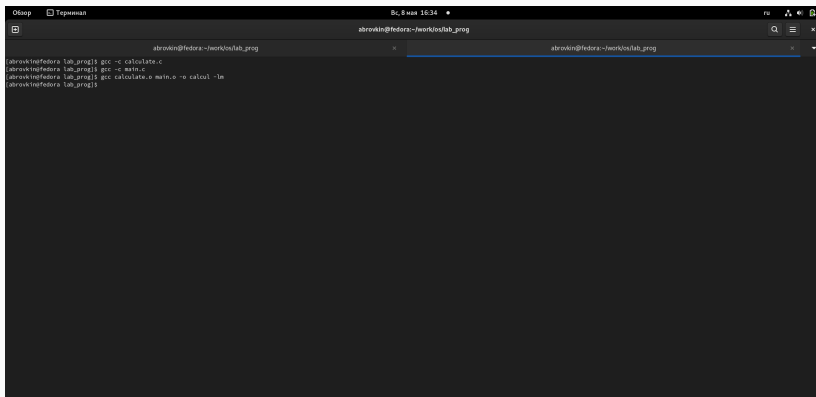
Основной файл main.c, реализующий интерфейс пользователя к калькулятору:



```
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("Введите: ");
11    scanf("%f", &Numeral);
12    printf("Операция: (+, -, *, /, pow, sqrt, sin, cos, tan): ");
13    scanf("%s", &Operation);
14    Result = Calculate(Numeral, Operation);
15    printf("Результат: %f", Result);
16    return 0;
17 }
```

Figure 6: Основной файл мэйн

3. Выполнил компиляцию программы посредством gcc:



The image shows a terminal window with a dark background. The title bar at the top indicates the window is titled 'Терминал' (Terminal) and shows the system time as 'Вт, 8 мая 16:34'. The terminal content shows a user named 'abrovkin' at a 'fedora' machine in the directory '~/work/os/lab_prog'. The user has executed three commands: 'gcc -o calculate -c', 'gcc -o main.c', and 'gcc calculate.o main.o -o calcul -lm'. The output of these commands is not visible in the screenshot.

```
abrovkin@fedora:~/work/os/lab_prog
abrovkin@fedora:~/work/os/lab_prog
abrovkin@fedora:~/work/os/lab_prog
[abrovkin@fedora lab_prog]$ gcc -o calculate -c
[abrovkin@fedora lab_prog]$ gcc -o main.c
[abrovkin@fedora lab_prog]$ gcc calculate.o main.o -o calcul -lm
[abrovkin@fedora lab_prog]$
```

Figure 7: Компиляция

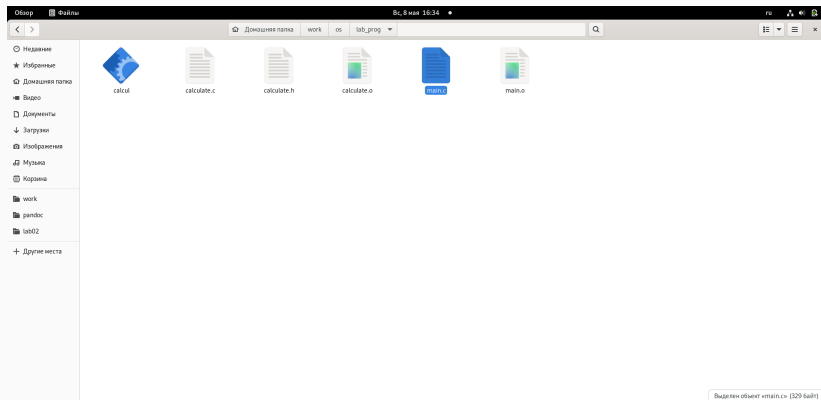


Figure 8: Файлы

4. Исправил синтаксические ошибки.
5. Создал Makefile

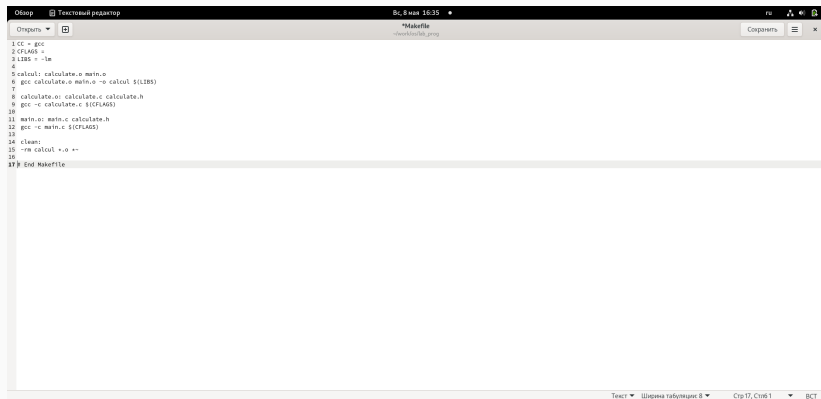


Figure 9: Создал Мэйкфайл

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

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

```
cc = gcc
CFLAGS = -g
LBS = -lm

calcul: calculate.o main.o
$(CC) calculate.o main.o -o calcul $(LBS)

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
```

Makefile All 112 (GNUmakefile)

Welcome to GNU Emacs, one component of the GNU/Linux operating system.

[Emacs Tutorial](#) Learn basic keystroke commands (read Emacs)

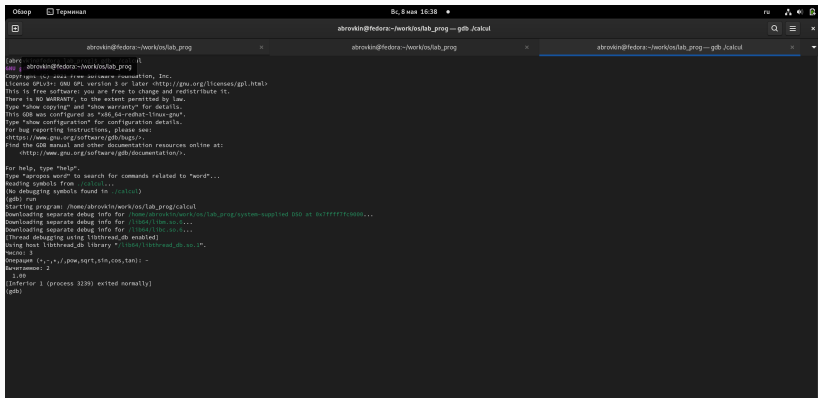
[Emacs Quickstart](#) Overview of Emacs features at gnu.org

[Emacs Emacs Manual](#) View the Emacs manual online

[Emacs Emacs](#) Top 10 (Fundamentals)

Wrote /home/abrovin/work/os/Lab_prog/Makefile

Figure 10: Исправил мэйкфайл



The screenshot shows a terminal window titled "Терминал" with the user "abrovkin@fedora" and the time "Вт, 6 мая 16:38". The terminal displays the output of the command `gdb ./calcul`. The output includes the GNU GPL license text, the path to the program `./calcul`, and the execution of the program. The program's output is as follows:

```
Starting program: /home/abrovkin/work/os/lab_prog/calcul
Downloading separate debug info for /home/abrovkin/work/os/lab_prog/system-supplied DSO at 8d7ffff7f0900...
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".
main: 3
stepsize (-1,-1,./,pow,sqrt,sin,cos,tan): -
iterations: 2
1.00
[Inferior 1 (process 3239) exited normally]
(gdb)
```

Figure 11: Запустил программу

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


```
Олеоп Терминал 8:6 мая 17:23
abrovkin@fedora:~/workos/lab_prog — gdb ./calcul

abrovkin@fedora:~/workos/lab_prog — gdb ./calcul

GNU gdb
Copyright (c) 2015 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) list
1      #include <stdio.h>
2      #include "calculata.h"
3
4      int
5      main (void)
6      {
7          float Numeral;
8          show Operation(4);
9          float Result;
10         printf "Result: ";
11
12         printf "Usage: %s [-v] [-pwm,wpf,sh,cos,tan]: ";
13         scanf "%s" Operation;
14         Result = Calculate Numeral Operation;
15         printf "Result: %f\n", Result;
(gdb) list calculate.c:20,25
16         return Numeral SecondNumeral;
17     }
18     else if (strcmp Operation,"*") == 0)
19     {
20         printf "Numerate: ";
21         scanf "%f" SecondNumeral;
22         return Numeral SecondNumeral;
23     }
24     else if (strcmp Operation,"/") == 0)
25     {
26         printf "Numerate: ";
27         scanf "%f" SecondNumeral;
28         return Numeral SecondNumeral;
29     }
30 }
```

Figure 13: Используя команды

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

```
for help, type 'help'.
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./calculate...
(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("Result: ");
(gdb) list 12,15
12     printf("Result: %f", (pow(2,10)+pow(2,10)+pow(2,10)));
13     scanf("%s", operation);
14     Result = Calculate(Numeral, operation);
15     printf("Result: ");
(gdb) list calculate.c:20,25
20     return Numeral * SecondNumeral;
21 }
22 else if (strcmp(operation, "x") == 0)
23 {
24     printf("Result: ");
25     scanf("%f", &SecondNumeral);
26     return Numeral * SecondNumeral;
27 }
28 else if (strcmp(operation, "x") == 0)
29 {
30 }
(gdb) list calculate.c:29,37
36     return Numeral * SecondNumeral;
37 }
38 else if (strcmp(operation, "x") == 0)
39 {
40 }
41 printf("Result: ");
42 scanf("%f", &SecondNumeral);
43 return Numeral * SecondNumeral;
44 }
(gdb) break 21
Breakpoint 1 at 00002021: File calculate.c, line 22.
(gdb)
```

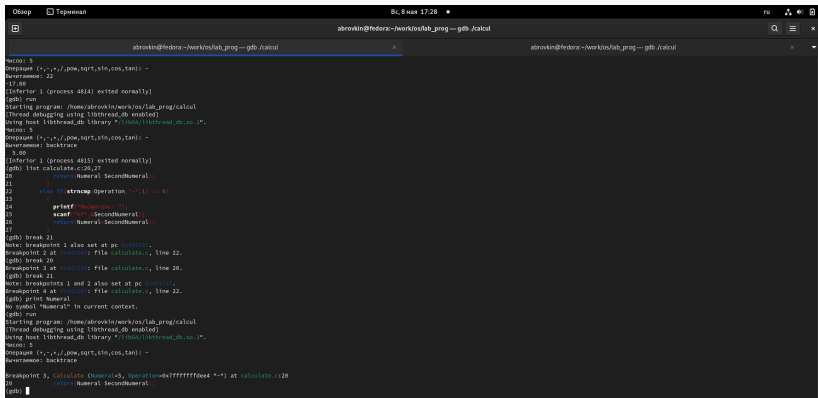
Figure 14: Параметры Лист

```
Ослож Терминал
Bc, 6 мая 17:29
abrovkin@fedora:~/work/os/lab_prog — gdb ./calcul

abrovkin@fedora:~/work/os/lab_prog — gdb ./calcul
(gdb) break 21
Note: breakpoint 1 also set at pc 0x401107.
Breakpoint 2 at 0x401107: File calculate.c, line 22.
(gdb) break 28
Breakpoint 3 at 0x401134: File calculate.c, line 26.
(gdb) break 21
Note: breakpoints 1 and 2 also set at pc 0x401107.
Breakpoint 4 at 0x401107: 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".
Numeral: 5
Breakpoint 1, 0x401107: File calculate.c, line 22.
Breakpoint 2, 0x401107: File calculate.c, line 22.
Breakpoint 3, 0x401134: File calculate.c, line 26.
Breakpoint 4, 0x401107: File calculate.c, line 22.
(gdb) print Numeral
Numeral = 5
(gdb) display Numeral
1: Numeral = 5
(gdb) info breakpoints
No defined info command: "breakpoints". Try "help info".
(gdb) info breakpoints
Num Type Disps Enb Address What
1 breakpoint keep y 0x0000000000000107 in calculate at calculate.c:22
2 breakpoint keep y 0x0000000000000128 in calculate at calculate.c:28
3 breakpoint keep y 0x0000000000000134 in calculate at calculate.c:26
4 breakpoint already hit 1 time
5 breakpoint keep y 0x0000000000000107 in calculate at calculate.c:22
(gdb) delete 1
(gdb) delete 2
(gdb) delete 3
(gdb) delete 4
(gdb)
```

Figure 15: Информация о точках останова

– запустил программу внутри отладчика и убедился, 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: – посмотрел, чему равно на этом этапе значение переменной `Numeral`, введя: `print Numeral` – сравнил с результатом вывода на экран после использования команды: `display Numeral` – убрал точки останова: `info breakpoints delete 1`



```
Олеор Терминал
Bc, 6 мая 17:28
abrovkin@fedora:~/workos/lab_prog — gdb ./calcul

abrovkin@fedora:~/workos/lab_prog — gdb ./calcul
(gdb) run
Starting program: /home/abrovkin/workos/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Secs: 5
Stepaway (+,-,*,/,pow,sqrt,sin,cos,tan): -
New runtime: 22
5.80
[Inferior 1 (process 4814) exited normally]
(gdb) list
20 return Numeral * SecondNumeral;
21
22 else if (strcmp (operation,"*") == 0)
23 {
24     printf ("Numerals: %i\n",
25           scanf ("%i%i", &SecondNumeral,
26                 &Numeral * SecondNumeral));
27 }
(gdb) break 21
Note: breakpoint 1 also set at pc 0x401200.
Breakpoint 2 at 0x401200: file calculate.c, line 22.
(gdb) break 20
Breakpoint 3 at 0x401200: file calculate.c, line 20.
(gdb) break 21
Note: breakpoints 1 and 2 also set at pc 0x401200.
Breakpoint 4 at 0x401200: file calculate.c, line 22.
(gdb) print Numeral
No symbol "Numeral" in current context.
(gdb) run
Starting program: /home/abrovkin/workos/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Secs: 5
Stepaway (+,-,*,/,pow,sqrt,sin,cos,tan): -
New runtime: 22
Breakpoint 3, calculate (Numeral=5, operation=0x7fffff000000) at calculate.c:20
20 return Numeral * SecondNumeral;
(gdb)
```

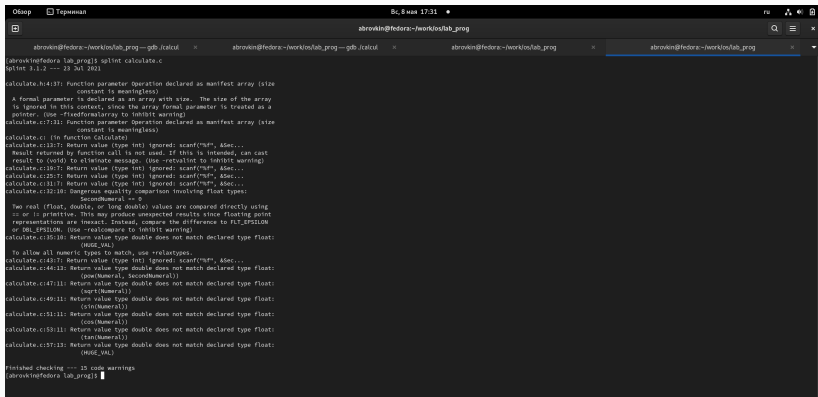
Figure 16: Запуск программы


```
Ослож Терминал
Bc, 6 мая 17:29
abrovkin@fedora:~/work/os/lab_prog — gdb ./calcul

abrovkin@fedora:~/work/os/lab_prog — gdb ./calcul
(gdb) break 21
Note: breakpoint 1 also set at pc 0x401107.
Breakpoint 2 at 0x401107: file calculate.c, line 22.
(gdb) break 28
Breakpoint 3 at 0x401134: file calculate.c, line 26.
(gdb) break 21
Note: breakpoints 1 and 2 also set at pc 0x401107.
Breakpoint 4 at 0x401107: 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".
Numeral: 5
Breakpoint 1, 0x401107: file calculate.c, line 22.
Breakpoint 2, 0x401107: file calculate.c, line 22.
Breakpoint 3, Calculate (Numeral=5, Operation=0x7fffffffdd44) at calculate.c:20
20     return Numeral * SecondNumeral;
(gdb) print Numeral
Numeral = 5
(gdb) display Numeral
1: Numeral = 5
(gdb) info breakpoints
No defined info command: "breakpoints". Try "help info".
(gdb) info breakpoints
Num  Type      Disposition Address      What
1  breakpoint keep y  0x0000000000001107 in calculate at calculate.c:22
2  breakpoint keep y  0x0000000000001134 in calculate at calculate.c:26
3  breakpoint keep y  0x0000000000001107 in calculate at calculate.c:22
4  breakpoint already hit 1 time
5  breakpoint keep y  0x0000000000001107 in calculate at calculate.c:22
(gdb) delete 1
(gdb) delete 2
(gdb) delete 3
(gdb) delete 4
(gdb)
```

Figure 18: Удаляю точки останова

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

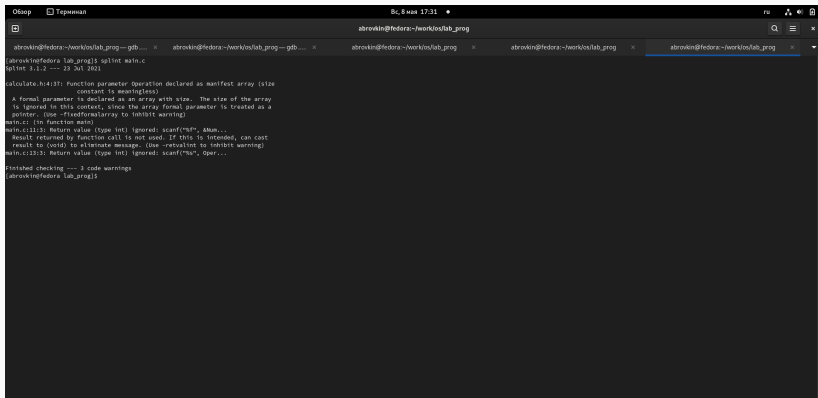


```
abrovkin@fedora:~/work/ios/lab_prog
abrovkin@fedora:~/work/ios/lab_prog$ splint calculate.c
splint 3.1.2 --- 23 Jul 2021

calculate.h:43:7: 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:73:1: Function parameter operation declared as manifest array (size
constant is meaningless)
calculate.c: (In function Calculate)
calculate.c:23:7: 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:30:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:25:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:31:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:32:10: Opaque equality comparison involving float types:
SecondNumerical == 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:35:10: Return value type double does not match declared type float:
(HUGE_VAL)
To allow all numeric types to match, use -relaxtypes.
calculate.c:43:7: Return value (type int) ignored: scanf("%d", &Sec...
calculate.c:44:12: Return value type double does not match declared type float:
(pow(Numerical, SecondNumerical))
calculate.c:47:11: Return value type double does not match declared type float:
(sqrt(Numerical))
calculate.c:49:11: Return value type double does not match declared type float:
(sin(Numerical))
calculate.c:51:11: Return value type double does not match declared type float:
(cos(Numerical))
calculate.c:53:11: Return value type double does not match declared type float:
(tan(Numerical))
calculate.c:57:13: Return value type double does not match declared type float:
(HUGE_VAL)

Finished checking --- 15 code warnings
abrovkin@fedora:~/work/ios/lab_prog$
```

Figure 19: splint



```
abrovkin@fedora:~/work/os/lab_prog --- gdb ... x abrovkin@fedora:~/work/os/lab_prog --- gdb ... x abrovkin@fedora:~/work/os/lab_prog x abrovkin@fedora:~/work/os/lab_prog x abrovkin@fedora:~/work/os/lab_prog x
abrovkin@fedora lab_prog$ splint main.c
splint 3.1.2 --- 23 Jul 2021

calculate.h:4137: 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:1 (in function main)
main.c:1113: 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:1233: Return value (type int) ignored: scanf("%s", &par...

Finished checking --- 3 code warnings
abrovkin@fedora lab_prog$
```

Figure 20: splint

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.