

Средства для создания приложений в ОС UNIX.

Эрик Кириллович Станиловский НПМбд-02-20¹

4 июня, 2021, Москва, Россия

¹Российский Университет Дружбы Народов

Цели и задачи работы

Цель лабораторной работы

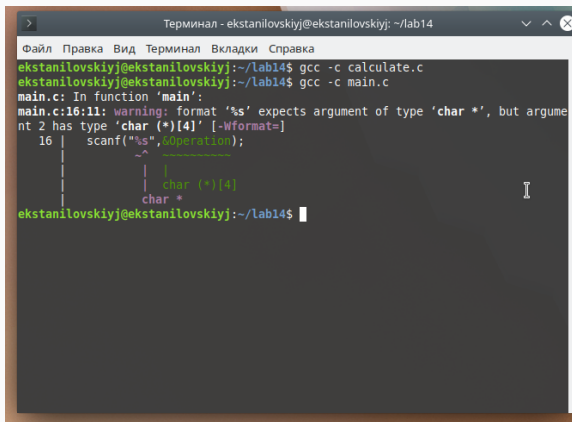
Приобрести простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.

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

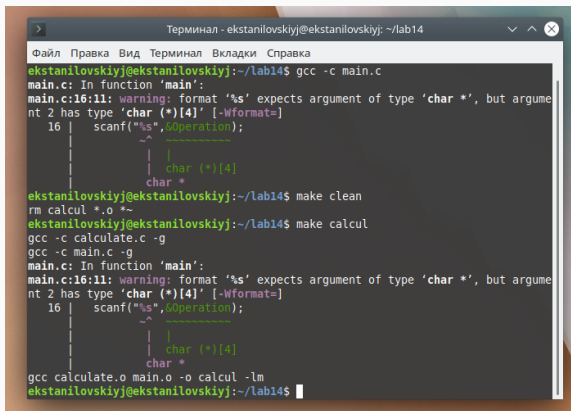
Выполнение работы



```
Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
ekstanilovskiyj@ekstanilovskiyj:~/lab14$ gcc -c calculate.c
ekstanilovskiyj@ekstanilovskiyj:~/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 *
ekstanilovskiyj@ekstanilovskiyj:~/lab14$
```

Figure 1: Компиляция

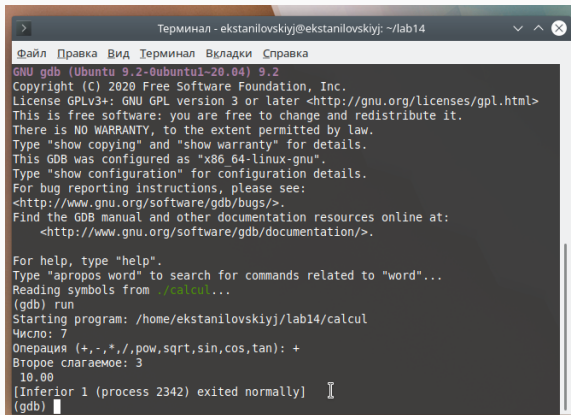
Выполнение работы



```
Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/lab14
Файл Правка Вид Терминал Вкладки Справка
ekstanilovskiyj@ekstanilovskiyj:~/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 *
ekstanilovskiyj@ekstanilovskiyj:~/lab14$ make clean
rm calcul *.o *~
ekstanilovskiyj@ekstanilovskiyj:~/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
ekstanilovskiyj@ekstanilovskiyj:~/lab14$
```

Figure 2: Использование make

Выполнение работы

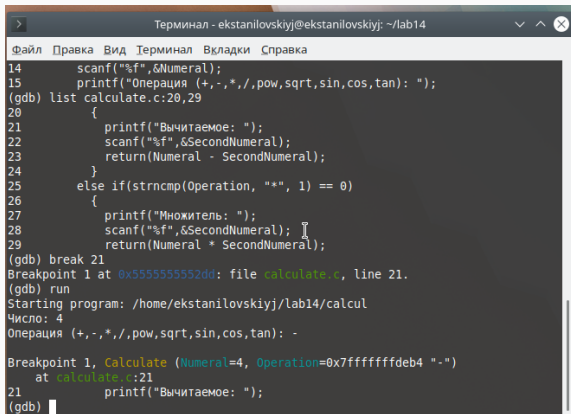
A screenshot of a terminal window titled "Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/lab14". The terminal shows the execution of the GDB debugger. It starts with the GDB version (9.2) and copyright information. The user enters the command "run", and the program "/home/ekstanilovskiyj/lab14/calcul" starts. The program prompts for an operation (+, -, *, /) and a second operand. The user enters "+" and "3". The program outputs "10.00". The terminal then shows "[Inferior 1 (process 2342) exited normally]" and "(gdb)".

```
Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/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/ekstanilovskiyj/lab14/calcul
Число: 7
Операция (+, -, *, /, pow, sqrt, sin, cos, tan): +
Второе слагаемое: 3
10.00
[Inferior 1 (process 2342) exited normally]
(gdb)
```

Figure 3: Использование отладчика

Выполнение работы

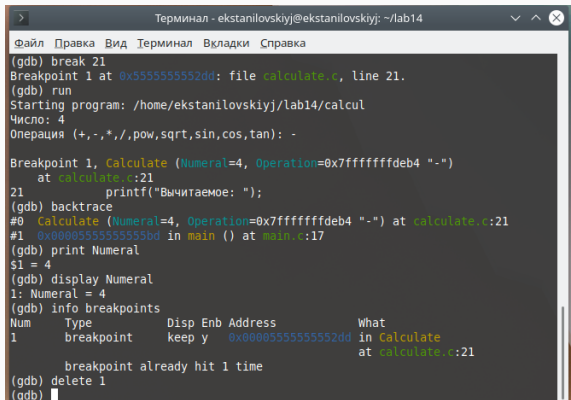


```
Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/lab14
Файл  Правка  Вид  Терминал  Вкладки  Справка
14      scanf("%f",&Numeral);
15      printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
(gdb) list calculate.c:20,29
20      {
21          printf("Вычитаемое: ");
22          scanf("%f",&SecondNumeral);
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/ekstanilovskiyj/lab14/calcul
Число: 4
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -

Breakpoint 1, Calculate (Numeral=4, Operation=0x7fffffffdeb4 "-")
at calculate.c:21
21      printf("Вычитаемое: ");
(gdb)
```

Figure 4: Использование отладчика

Выполнение работы

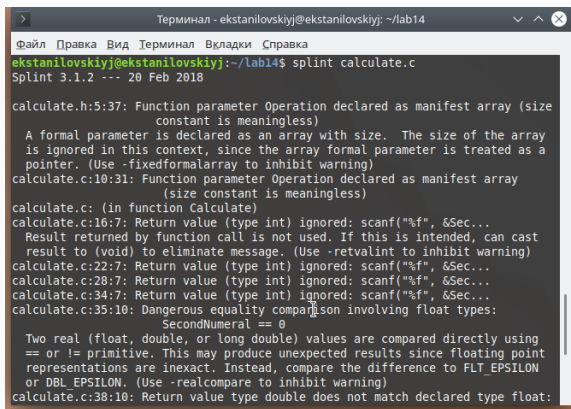


```
Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/lab14
Файл Правка Вид Терминал Вкладки Справка
(gdb) break 21
Breakpoint 1 at 0x555555552dd: file calculate.c, line 21.
(gdb) run
Starting program: /home/ekstanilovskiyj/lab14/calcul
Число: 4
Операция (+, -, *, /, pow, sqrt, sin, cos, tan): -

Breakpoint 1, calculate (Numeral=4, Operation=0x7fffffffdeb4 "-")
at calculate.c:21
21      printf("Вычитаемое: ");
(gdb) backtrace
#0 calculate (Numeral=4, Operation=0x7fffffffdeb4 "-") at calculate.c:21
#1 0x00005555555555bd in main () at main.c:17
(gdb) print Numeral
$1 = 4
(gdb) display Numeral
1: Numeral = 4
(gdb) info breakpoints
Num      Type      Disp Enb Address          What
1        breakpoint keep y  0x0000555555552dd in calculate
                                                at calculate.c:21
breakpoint already hit 1 time
(gdb) delete 1
(gdb)
```

Figure 5: Использование отладчика

Выполнение работы



```
Терминал - ekstanilovskiyj@ekstanilovskiyj: ~/lab14
Файл Правка Вид Терминал Вкладки Справка
ekstanilovskiyj@ekstanilovskiyj:~/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.