

презентация по лабораторной работе 13

Markdown

Хрусталеv В.Н.

Российский университет дружбы народов, Москва, Россия

Информация

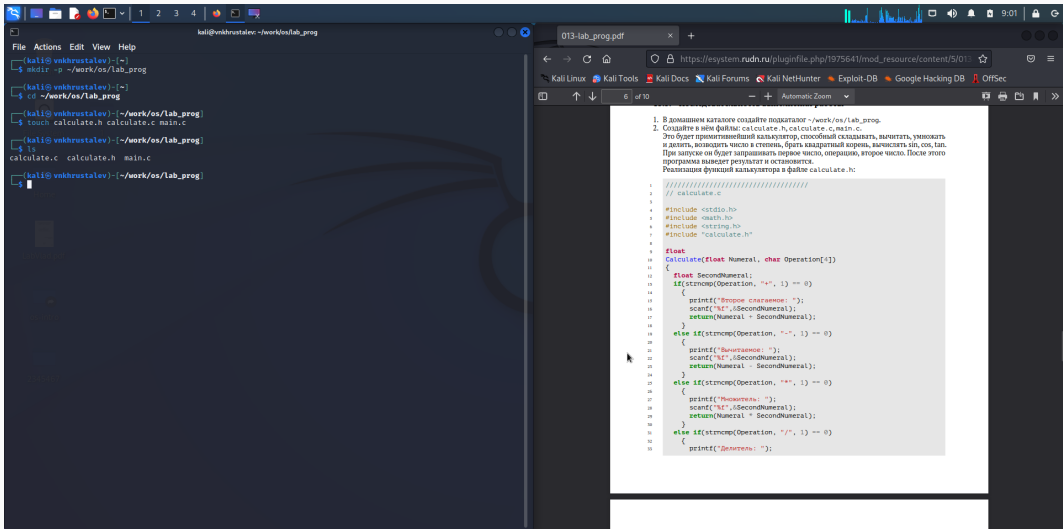
- Хрусталев Влад Николаевич
- Студент ФМиЕН РУДН
- Группа НПИбд-02-22

Вводная часть

- 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 работы

В домашнем каталоге создадим подкаталог `~/work/os/lab_prog`. После чего создадим в нём файлы: `calculate.h`, `calculate.c`, `main.c` и выполним проверку. Перейдём в `mousepad`.



В mousepad откроем созданный файл calculate.c и приступим к переносу в него скрипта из файла

The image shows a Kali Linux desktop environment. On the left, a Mousepad text editor window titled "-/workos/lab_prog/calculate.c - Mousepad" displays the C source code for a calculator. The code handles various arithmetic operations: addition, subtraction, multiplication, division, power, square root, sine, cosine, and tangent. It includes error handling for division by zero and invalid operations. On the right, a web browser window titled "013-lab_prog.pdf" shows the same code as a PDF document. Below the code in the PDF, there are two paragraphs of Russian text. The first paragraph states that the file calculate.h describes the function call format for the calculator interface. The second paragraph states that the main.c file implements the user interface for the calculator. The browser's address bar shows the URL "https://esystem.rudn.ru/pluginfile.php/1975641/mod_resource/content/5/013...". The browser's tab bar shows several open tabs including "Kali Linux", "Kali Tools", "Kali Docs", "Kali Forums", "Kali NetHunter", "Exploit-DB", "Google Hacking DB", and "OffSec". The browser's status bar shows "7 of 10" and "Automatic Zoom".

```
19 else if(strcmp(Operation, "-", 1) == 0)
20 {
21     printf("Вычитаемое: ");
22     scanf("%f", &SecondNumeral);
23     return(Numeral - SecondNumeral);
24 }
25 else if(strcmp(Operation, "*", 1) == 0)
26 {
27     printf("Умножитель: ");
28     scanf("%f", &SecondNumeral);
29     return(Numeral * SecondNumeral);
30 }
31 else if(strcmp(Operation, "/", 1) == 0)
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(strcmp(Operation, "pow", 3) == 0)
44 {
45     printf("Степень: ");
46     scanf("%f", &SecondNumeral);
47     return(pow(Numeral, SecondNumeral));
48 }
49 else if(strcmp(Operation, "sqrt", 4) == 0)
50     return(sqrt(Numeral));
51 else if(strcmp(Operation, "sin", 3) == 0)
52     return(sin(Numeral));
53 else if(strcmp(Operation, "cos", 3) == 0)
54     return(cos(Numeral));
55 else if(strcmp(Operation, "tan", 3) == 0)
56     return(tan(Numeral));
57 else
58 {
59     printf("Неправильно введено действие ");
60     return(HUGE_VAL);
61 }
62 }
63
```

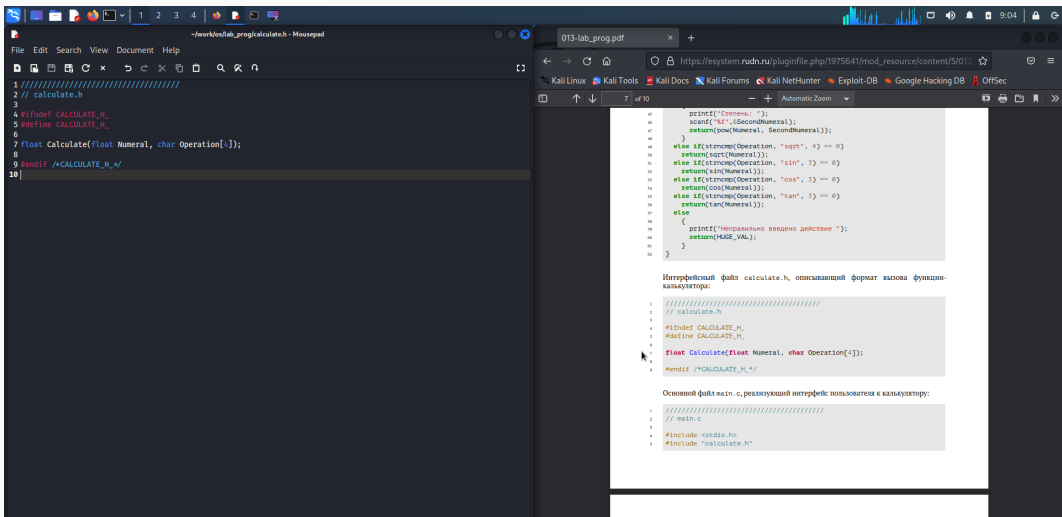
Интерфейсный файл calculate.h, описывающий формат вызова функции-калькулятора:

```
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*/
```

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

```
1 //////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
```


После того как мы перенесли и сохранили скрипт для первого файла, открываем файл `calculate.h` и также переносим в него скрипт, но уже для второго файла. Выполняем сохранение



```
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_*/
10
```

013-lab_prog.pdf

https://esystem.rudn.ru/pluginfile.php/1975641/mod_resource/content/5/013-lab_prog.pdf

```
45 printf("Сценарий: ");
46 scanf("%f", &SecondNumeral);
47 return(pow(Numeral, SecondNumeral));
48 }
49 else if(strcmp(Operation, "sqrt", 4) == 0)
50 return(sqrt(Numeral));
51 else if(strcmp(Operation, "sin", 3) == 0)
52 return(sin(Numeral));
53 else if(strcmp(Operation, "cos", 3) == 0)
54 return(cos(Numeral));
55 else if(strcmp(Operation, "tan", 3) == 0)
56 return(tan(Numeral));
57 else
58 {
59 printf("Неправильно введено действие ");
60 return(HUGE_VAL);
61 }
62 }
```

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

```
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_*/
```

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

```
1 //////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
5 #include "calculate.h"
```

Теперь нам нужно перенести последний третий скрипт в файл main.c. После чего также выполняем сохранение и закрываем emacs

The image shows a Kali Linux desktop environment. On the left, the Emacs text editor is open, displaying the contents of the file `main.c`. The code is as follows:

```
1 //////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
5 #include "calculate.h"
6
7 int
8 main (void)
9 {
10     float Numeral;
11     char Operation[4];
12     float Result;
13     printf("Число: ");
14     scanf("%f",&Numeral);
15     printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
16     scanf("%s",&Operation);
17     Result = Calculate(Numeral, Operation);
18     printf("%6.2f\n",Result);
19     return 0;
20 }
21
```

On the right, a web browser window is open, displaying a PDF document titled "013-lab_prog.pdf". The document content includes the following text:

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

```
1 //////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
5 #include "calculate.h"
```

Below this, the document lists the tasks for the lab:

3. Выполните компиляцию программы посредством gcc:

```
1 gcc -c calculate.c
2 gcc -c main.c
3 gcc calculate.o main.o -o calcul -lm
```

4. При необходимости исправляйте синтаксические ошибки.
5. Создайте Makefile со следующим содержанием:

```
1 #
2 # Makefile
3 #
```

В терминале выполним компиляцию программы посредством gcc

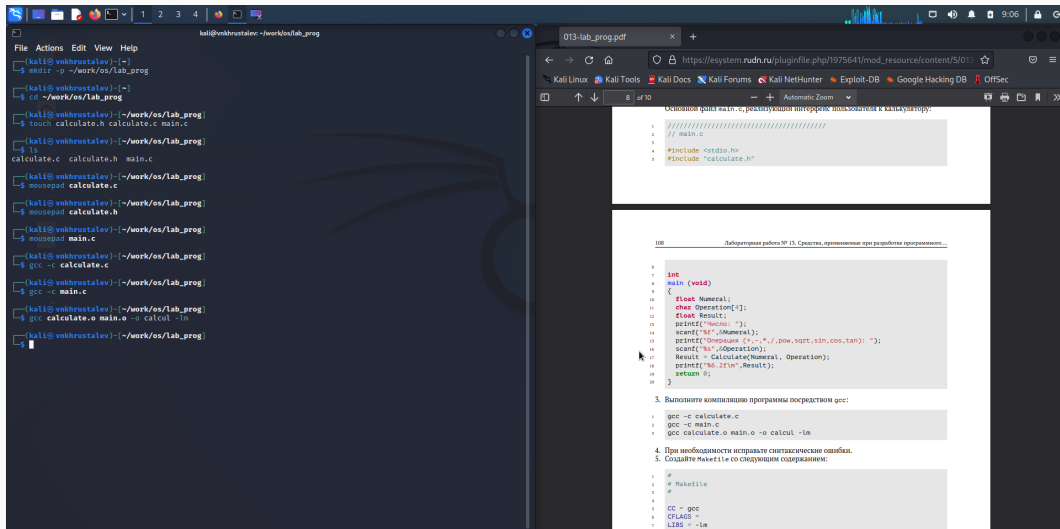


Рис. 5: Компиляция программы

Создадим Makefile и внесём туда небольшие изменения. В переменную CFLAGS добавил опцию -g, необходимую для компиляции объектных файлов и их использования в программе отладчика GDB. Сделал так, что утилита компиляции выбирается с помощью переменной CC

The screenshot shows a Kali Linux desktop environment. On the left, a text editor (Mousepad) displays a Makefile. On the right, a web browser (Chromium) shows a page with a C program and instructions for compilation and debugging.

Makefile Content:

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

C Program Content:

```
10 float Numeral;
11 char Operation[4];
12 float Result;
13 printf("Введите: ");
14 scanf("%f",&Numeral);
15 printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
16 scanf("%s",&Operation);
17 Result = Calculate(Numeral, Operation);
18 printf("%6.2f\n",Result);
19 return 0;
20 }
```

Instructions:

3. Выполните компиляцию программы посредством gcc:
4. При необходимости исправьте синтаксические ошибки.
5. Создайте Makefile со следующим содержанием:

Makefile Content:

```
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
```

Additional Instructions:

Понесите в отчёте его содержание.

6. С помощью gdb выполните отладку программы calcul (перед использованием gdb исправьте Makefile):
 - Запустите отладчик GDB, загрузив в него программу для отладки:

С помощью gdb выполним отладку программы calcul. После чего запустим программу командой run

The image shows a Kali Linux terminal window on the left and a web browser window on the right. The terminal window displays the execution of the GDB debugger on a program named 'calcul'. The browser window shows a document titled '013-lab_prog.pdf' with instructions for debugging.

Terminal Output:

```
kali@vnikhrustalev: ~/work/os/lab_prog
$ gdb ./calcul
GNU gdb (Debian 12.1-2) 13.1
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-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) run
Starting program: /home/kali/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Множитель: 1[C
0.00
[Inferior 1 (process 14818) exited normally]
(gdb) run
Starting program: /home/kali/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Множитель: 6
30.00
[Inferior 1 (process 15121) exited normally]
(gdb)
```

Web Browser Content (013-lab_prog.pdf):

Курсовое З.С.Ор. Операционные системы 109

- 1. `gdb ./calcul`
- Для запуска программы внутри отладчика введите команду `run`:
- 1. `run`
- Для пошагового (по 9 строк) просмотра исходного кода используйте команду `list`:
- 1. `list`
- Для просмотра строк с 12 по 15 основного файла используйте `list` с параметрами:
- 1. `list 12,15`
- Для просмотра определённых строк не основного файла используйте `list` с параметрами:
- 1. `list calculate.c:20,29`
- Установите точку останова в файле `calculate.c` на строке номер 21:
- 1. `list calculate.c:20,27`
- 1. `break 21`
- Выведите информацию об имеющихся в проекте точке останова:
- 1. `info breakpoints`
- Запустите программу внутри отладчика и убедитесь, что программа остановится в момент прохождения точки останова:
- 1. `run`
- 1. `5`
- 1. `backtrace`
- Отладчик выдает следующую информацию:
- 1. `#0 Calculate (Numerical=5, Operation=0x7fffffd280 "-")`
- 1. `at calculate.c:21`
- 1. `47: ~~~~~`

Воспользовавшись утилитой splint проанализируем коды файлов calculate.c и main.c. С помощью утилиты splint выяснилось, что в файлах calculate.c и main.c присутствует функция чтения scanf, возвращающая целое число (тип int), но эти числа не используются и нигде не сохраняются. Утилита вывела предупреждение о том, что в файле calculate.c происходит сравнение вещественного числа с нулем. Также возвращаемые значения (тип double) в функциях pow, sqrt, sin, cos и tan записываются в переменную типа float, что свидетельствует о потере данных

The screenshot shows a Kali Linux terminal window on the left and a web browser window on the right. The terminal window displays the output of the `splint` utility analyzing `calculate.c` and `main.c`. The browser window shows a document titled "013-lab_prog.pdf" from the URL `https://esystem.rudn.ru/pluginfile.php/1975641/mod_resource/content/5/013-lab_prog.pdf`. The document contains a lab assignment for a C programming course.

Terminal Output:

```
kali@vnkhkrustalev: ~/work/os/lab_prog
File Actions Edit View Help

(kali@vnkhkrustalev) [-~/work/os/lab_prog]
$ splint calculate.c
Splint 3.1.2 — 21 Feb 2021

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)
calculate.c:10:11: 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:
(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:
```

Browser Document Content:

Лабораторная работа № 13. Средства, применяемые при разработке программного ...

```
1 print Numeral
```

На экран должно быть выведено число 5.
Сравните с результатом вывода на экран после использования команды:

```
1 display Numeral
```

Уберите точки останова:

```
1 info breakpoints
2 delete 1
```

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

13.4. Содержание отчёта

1. Типовый лист с указанием номера лабораторной работы и ФИО студента.

Итоги

- В ходе выполнения лабораторной работы мы 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