

Лабораторная работа 13

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

Паласиос Ф.

05 мая 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Паласиос Фелипе
- студент группы НКАбд - 04 - 22
- Российский университет дружбы народов

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.

1. В домашнем каталоге создайте подкаталог `~/work/os/lab_prog`.
2. Создайте в нём файлы: `calculate.h`, `calculate.c`, `main.c`. Это будет примитивнейший калькулятор, способный складывать, вычитать, умножать и делить, возводить число в степень, брать квадратный корень, вычислять `sin`, `cos`, `tan`. При запуске он будет запрашивать первое число, операцию, второе число. После этого программа выведет результат и остановится. Реализация функций калькулятора в файле `calculate.h`:

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

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

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

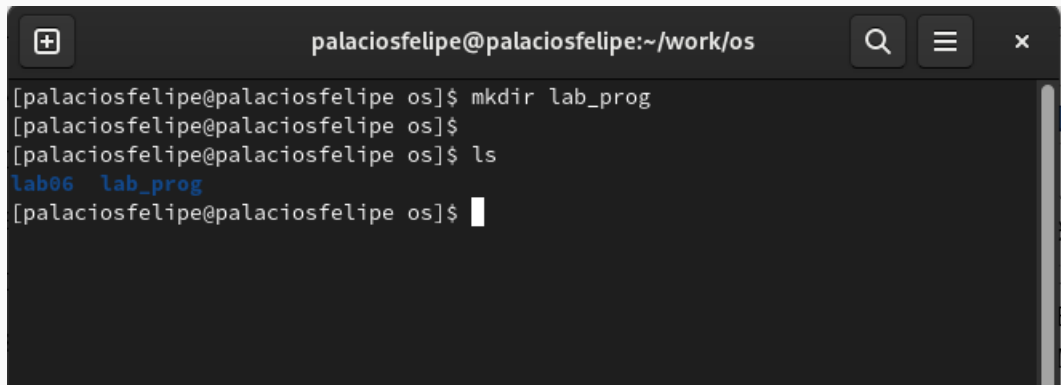
6. С помощью gdb выполните отладку программы calcul (перед использованием gdb исправьте Makefile):
 - Запустите отладчик GDB, загрузив в него программу для отладки –
 - Для запуска программы внутри отладчика введите команду run: –
 - Для постраничного (по 9 строк) просмотра исходного код используйте команду list: –
 - Для просмотра строк с 12 по 15 основного файла используйте list с параметрами:
 - Для просмотра определённых строк не основного файла используйте list с па- раметрами

- Установите точку останова в файле `calculate.c` на строке номер 21:
- Выведите информацию об имеющихся в проекте точка останова:
- Запустите программу внутри отладчика и убедитесь, что программа остановится в момент прохождения точки останова:
- Отладчик выдаст следующую информацию: а команда `backtrace` покажет весь стек вызываемых функций от начала программы до текущего места.

- Посмотрите, чему равно на этом этапе значение переменной `Numeral`, введя: На экран должно быть выведено число 5.
 - Сравните с результатом вывода на экран после использования команды:
 - Уберите точки останова.
7. С помощью утилиты `splint` попробуйте проанализировать коды файлов `calculate.c` и `main.c`

Выполнение лабораторной работы

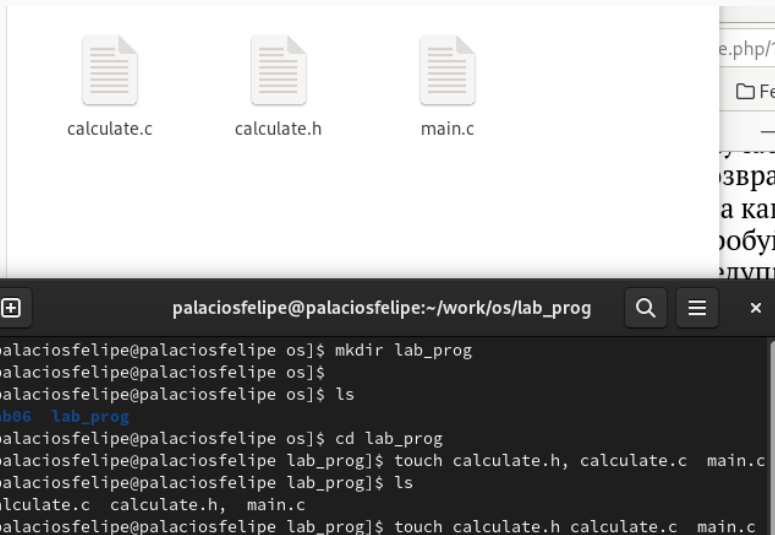
1. В домашнем каталоге создайте подкаталог ~/work/os/lab_prog.

A terminal window with a dark background. The title bar shows the user 'palaciosfelipe' and the current directory '~/work/os'. The terminal contains the following text:

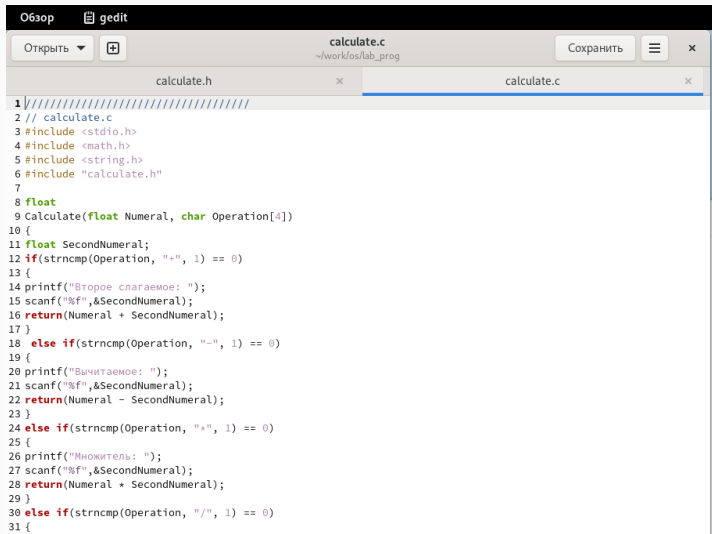
```
[palaciosfelipe@palaciosfelipe os]$ mkdir lab_prog  
[palaciosfelipe@palaciosfelipe os]$  
[palaciosfelipe@palaciosfelipe os]$ ls  
lab06  lab_prog  
[palaciosfelipe@palaciosfelipe os]$
```

Рис. 1: lab_prog

2. Создайте в нём файлы: calculate.h, calculate.c, main.c.

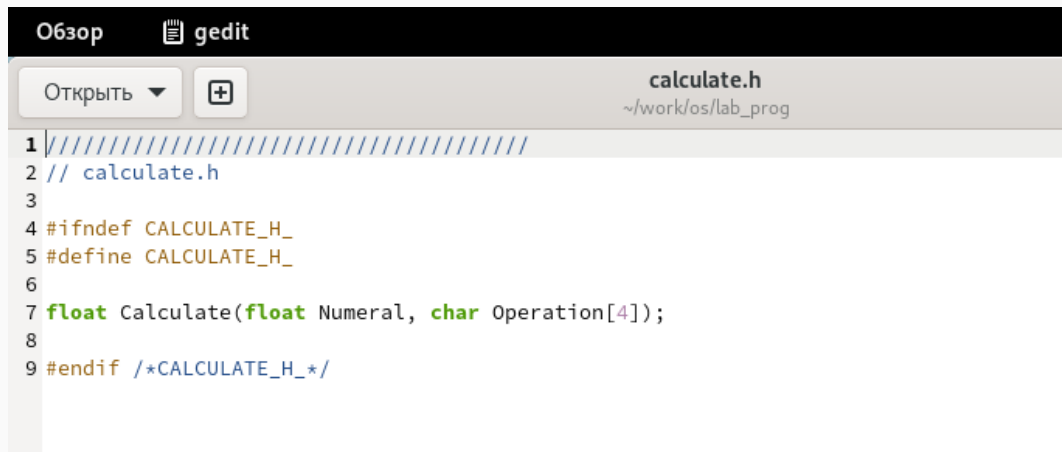


Реализация функций калькулятора в файле calculate.c



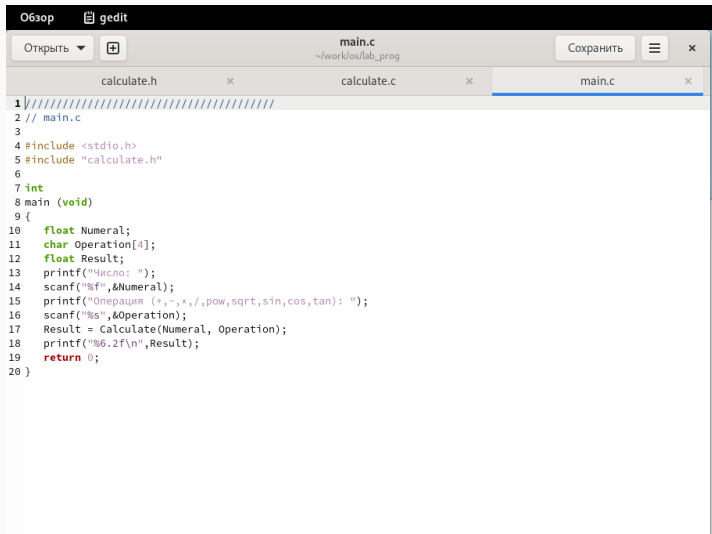
```
Обзор gedit
calculate.c
~/work/os/lab_prog
Сохранить x
calculate.h x calculate.c x
1 //////////////////////////////////////////////////
2 // calculate.c
3 #include <stdio.h>
4 #include <math.h>
5 #include <string.h>
6 #include "calculate.h"
7
8 float
9 Calculate(float Numeral, char Operation[4])
10 {
11     float SecondNumeral;
12     if(strncmp(Operation, "+", 1) == 0)
13     {
14         printf("Второе слагаемое: ");
15         scanf("%f",&SecondNumeral);
16         return(Numeral + SecondNumeral);
17     }
18     else if(strncmp(Operation, "-", 1) == 0)
19     {
20         printf("Вычитаемое: ");
21         scanf("%f",&SecondNumeral);
22         return(Numeral - SecondNumeral);
23     }
24     else if(strncmp(Operation, "*", 1) == 0)
25     {
26         printf("Множитель: ");
27         scanf("%f",&SecondNumeral);
28         return(Numeral * SecondNumeral);
29     }
30     else if(strncmp(Operation, "/", 1) == 0)
31     {
```

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



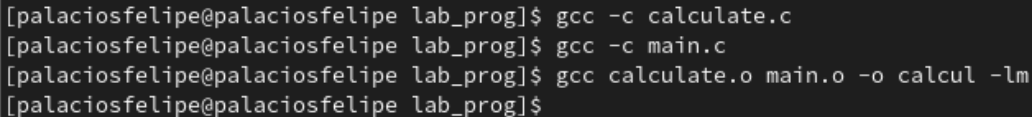
```
1 //////////////////////////////////////////////////
2 // calculate.h
3
4 #ifndef CALCULATE_H_
5 #define CALCULATE_H_
6
7 float Calculate(float Numeral, char Operation[4]);
8
9 #endif /*CALCULATE_H_*/
```

Основной файл main.c, реализующий интерфейс пользователя к калькулятору



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


3. Выполните компиляцию программы посредством gcc:

A terminal window with a dark background and light gray text. It shows four lines of commands and their prompts. The first line is '[palaciosfelipe@palaciosfelipe lab_prog]\$ gcc -c calculate.c'. The second line is '[palaciosfelipe@palaciosfelipe lab_prog]\$ gcc -c main.c'. The third line is '[palaciosfelipe@palaciosfelipe lab_prog]\$ gcc calculate.o main.o -o calcul -lm'. The fourth line is '[palaciosfelipe@palaciosfelipe lab_prog]\$' with no command entered.

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

Рис. 6: Компиляция

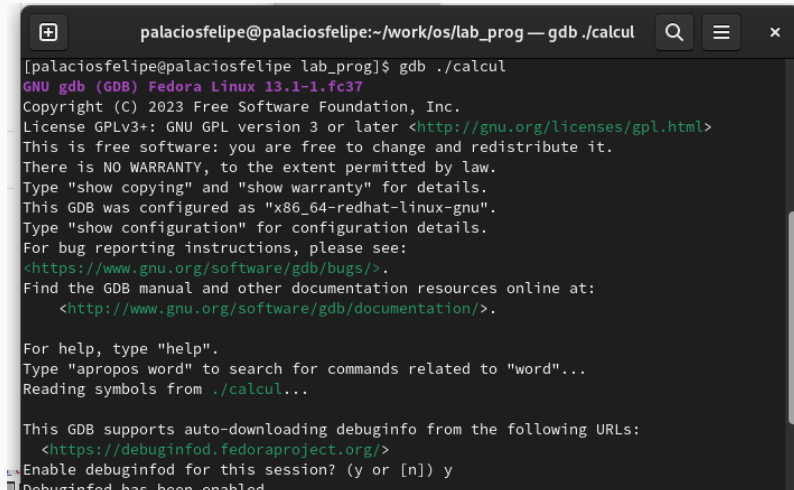
4. При необходимости исправьте синтаксические ошибки.
5. Создайте Makefile со следующим содержанием:



The screenshot shows a code editor window titled "Makefile" with the path "~/work/os/lab_prog". The editor has tabs for "presentation.md", "calculate.c", "calculate.h", "main.c", and "Makefile". The "Makefile" tab is active, displaying the following content with red squiggly lines indicating syntax errors:

```
#  
# 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)
```

6. С помощью gdb выполните отладку программы calcul (перед использованием gdb исправьте Makefile)



```
palaciosfelipe@palaciosfelipe:~/work/os/lab_prog — gdb ./calcul
[palaciosfelipe@palaciosfelipe lab_prog]$ gdb ./calcul
GNU gdb (GDB) Fedora Linux 13.1-1.fc37
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-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...

This GDB supports auto-downloading debuginfo from the following URLs:
  <https://debuginfod.fedoraproject.org/>
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
```

```
palaciosfelipe@palaciosfelipe:~/work/os/lab_prog — gdb ./calcul

To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Downloading separate debug info for /home/palaciosfelipe/work/os/lab_prog/calcul
(No debugging symbols found in ./calcul)
(gdb) run
Starting program: /home/palaciosfelipe/work/os/lab_prog/calcul
Downloading separate debug info for system-supplied DSO at 0x7ffff7fc6000
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: 20
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -
Вычитаемое: 10
10.00
[Inferior 1 (process 7357) exited normally]
(gdb) 50
Undefined command: "50". Try "help".
(gdb) run
Starting program: /home/palaciosfelipe/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 50
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Множитель: 20
1000.00
[Inferior 1 (process 7403) exited normally]
(gdb)
```

```
palaciosfelipe@palaciosfelipe:~/work/os/lab_prog — gdb ./calcul

Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Му Множитель: 10
100.00
Ко [Inferior 1 (process 7762) exited normally]
(gdb) list
1 //////////////////////////////////////////////////
2 // main.c
3
4 #include <stdio.h>
5 #include "calculate.h"
6
7 int
8 main (void)
9 {
10     float Numeral;
(gdb) list
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("%.2f\n",Result);
19     return 0;
20 }
(gdb)
```

```
palaciosfelipe@palaciosfelipe:~/work/os/lab_prog — gdb ./calcul

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

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

Рис. 12: list 20,29

```
(gdb) list calculate.c:20,27
20     printf("Вычитаемое: ");
21     scanf("%f",&SecondNumeral);
22     return(Numeral - SecondNumeral);
23 }
24 else if(strncmp(Operation, "*", 1) == 0)
25 {
26     printf("Множитель: ");
27     scanf("%f",&SecondNumeral);
(gdb) break 21
Breakpoint 1 at 0x40121e: file calculate.c, line 21.
(gdb) █
```

Рис. 13: list 20,27, break 21


```
(gdb) info breakpoints
Num      Type           Disp Enb Address                  What
1        breakpoint    keep y   0x00000000000040121e in Calculate
                                     at calculate.c:21
(gdb) 
```

Рис. 14: info breakpoints

```
(gdb) run
Starting program: /home/palaciosfelipe/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): -

Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffffdf34 "-")
    at calculate.c:21
21      scanf("%f",&SecondNumeral);
(gdb) backtrace
#0 Calculate (Numeral=5, Operation=0x7fffffffdf34 "-") at calculate.c:21
#1 0x00000000004014eb in main () at main.c:17
(gdb)
```

Рис. 15: backtrace

```
#1  0x00000000004014eb in main () at main.c:17  
(gdb) print Numeral  
$1 = 5  
(gdb)
```

Рис. 16: print Numeral

```
(gdb) print Numeral  
$1 = 5  
(gdb) display Numeral  
1: Numeral = 5  
(gdb) █
```

Рис. 17: display Numera

```
(gdb) info breakpoints
Num      Type           Disp Enb Address              What
1        breakpoint     keep y   0x0000000000004012le in Calculate
                                at calculate.c:21
c
    breakpoint already hit 1 time
(gdb) delete 1
(gdb)
```

Рис. 18: info breakpoints delete 1

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

```
[palaciosfelipe@palaciosfelipe lab_prog]$ splint calculate.c
\bash: splint: команда не найдена...
Установить пакет «splint», предоставляющий команду «splint»? [N/y] y

* Ожидание в очереди...
Следующие пакеты должны быть установлены:
splint-3.1.2-29.fc37.x86_64    An implementation of the lint program
Продолжить с этими изменениями? [N/y] y

* Ожидание в очереди...
* Ожидание аутентификации...
* Ожидание в очереди...
* Загрузка пакетов...
```

```
[palaciosfelipe@palaciosfelipe lab_prog]$ splint main.c
Splint 3.1.2 --- 23 Jul 2022

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)
main.c: (in function main)
main.c:14:4: 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:16:15: 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:16:12: Corresponding format code
main.c:16:4: Return value (type int) ignored: scanf("%s", &Ope...

Finished checking --- 4 code warnings
[palaciosfelipe@palaciosfelipe lab_prog]$
```

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.