

Лабораторная работа №13

Шубнякова Дарья, НКАбд-03-22

1. Цель
2. Теоретическое введение
3. Основные задачи
4. Процесс выполнения
5. Вывод
6. Список литературы

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.

Процесс разработки программного обеспечения обычно разделяется на следующие этапы:

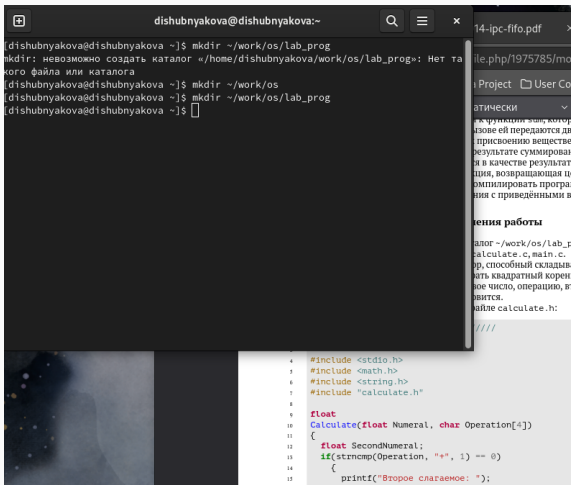
- планирование, включающее сбор и анализ требований к функционалу и другим характеристикам разрабатываемого приложения;
- проектирование, включающее в себя разработку базовых алгоритмов и спецификаций,
- определение языка программирования;
- непосредственная разработка приложения;

- кодирование — по сути создание исходного текста программы (возможно в нескольких вариантах);
- анализ разработанного кода;
- сборка, компиляция и разработка исполняемого модуля;
- тестирование и отладка, сохранение произведённых изменений;
- документирование.

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

1. Создаем подкаталог lab_prog в рабочем каталоге.



The image shows a terminal window and a code editor. The terminal window, titled 'dishubnyakova@dishubnyakova:~', shows the following commands and output:

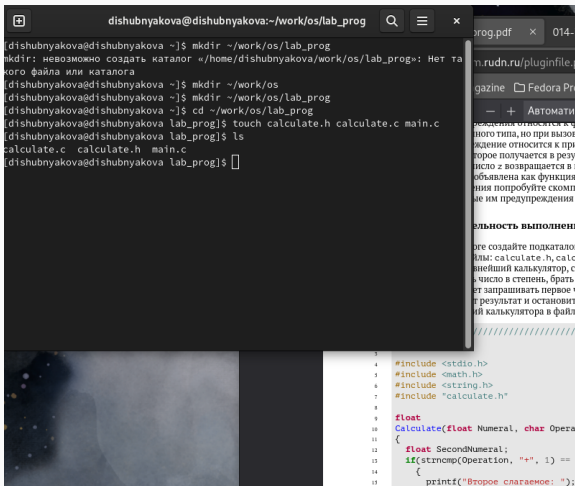
```
[dishubnyakova@dishubnyakova ~]$ mkdir ~/work/os/lab_prog
mkdir: невозможно создать каталог «/home/dishubnyakova/work/os/lab_prog»: Нет тако-
го файла или каталога
[dishubnyakova@dishubnyakova ~]$ mkdir ~/work/os
[dishubnyakova@dishubnyakova ~]$ mkdir ~/work/os/lab_prog
[dishubnyakova@dishubnyakova ~]$
```

The code editor shows a C program named 'calculate.h' and 'calculate.c'. The code includes standard headers and defines a function 'Calculate' that takes a float and a string operation, and returns a float. The code is as follows:

```
1 //
2
3
4 #include <stdio.h>
5 #include <math.h>
6 #include <string.h>
7 #include "calculate.h"
8
9 float
10 Calculate(float Numeral, char Operation[4])
11 {
12     float SecondNumeral;
13     if(strcmp(Operation, "+", 1) == 0)
14     {
15         printf("Второе слагаемое: ");
16         //
17     }
18 }
```

Рис. 1: Создание рабочего каталога

2. Создаем три необходимых для дальнейшей работы файла: calculate.h, calculate.c, main.c.



The screenshot shows a terminal window with the following commands and output:

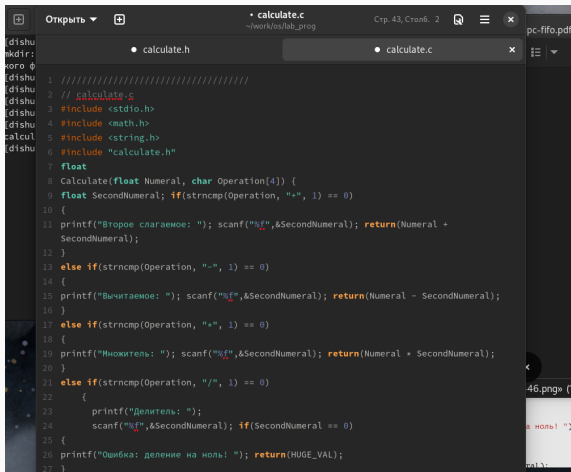
```
dishubnyakova@dishubnyakova:~/work/os/lab_prog
[dishubnyakova@dishubnyakova ~]$ mkdir ~/work/os/lab_prog
mkdir: невозможно создать каталог «/home/dishubnyakova/work/os/lab_prog»: Нет та
кого файла или каталога
[dishubnyakova@dishubnyakova ~]$ mkdir ~/work/os
[dishubnyakova@dishubnyakova ~]$ mkdir ~/work/os/lab_prog
[dishubnyakova@dishubnyakova ~]$ cd ~/work/os/lab_prog
[dishubnyakova@dishubnyakova lab_prog]$ touch calculate.h calculate.c main.c
[dishubnyakova@dishubnyakova lab_prog]$ ls
calculate.c calculate.h main.c
[dishubnyakova@dishubnyakova lab_prog]$
```

Below the terminal window, a code editor shows the content of main.c:

```
1
2
3
4 #include <stdio.h>
5 #include <math.h>
6 #include <string.h>
7 #include "calculate.h"
8
9 float
10 Calculate(float Numeral, char Opera
11 {
12     float SecondNumeral;
13     if(strcmp(Operation, "+") ==
14     {
15         printf("Второе слагаемое: ");
```

Рис. 3: Создание файлов

3. Копируем реализацию функций калькулятора в calculate.c.

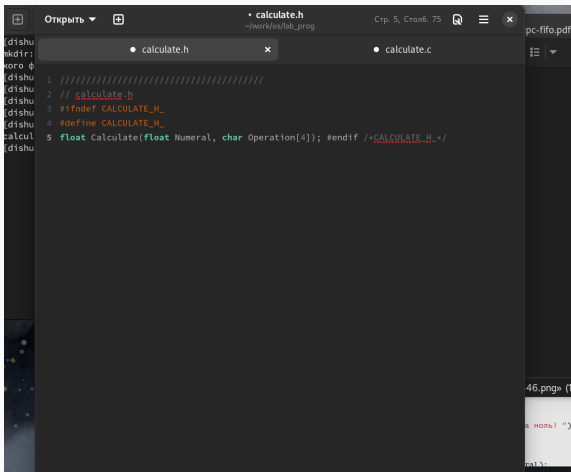


The screenshot shows a code editor with a dark theme. The active file is 'calculate.c' at line 43, column 6. The code implements a function 'Calculate' that takes a float 'Numeral' and a char array 'Operation' of size 4. It uses 'scanf' to read a second number and 'return' to provide the result based on the operation. The operations supported are addition (+), subtraction (-), multiplication (*), and division (/). Division by zero is handled by returning 'HUGE_VAL' and printing an error message.

```
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     float SecondNumeral; if(strcmp(Operation, "+", 1) == 0)
11     {
12         printf("Второе слагаемое: "); scanf("%f",&SecondNumeral); return(Numeral +
13         SecondNumeral);
14     }
15     else if(strcmp(Operation, "-", 1) == 0)
16     {
17         printf("Вычитаемое: "); scanf("%f",&SecondNumeral); return(Numeral - SecondNumeral);
18     }
19     else if(strcmp(Operation, "*", 1) == 0)
20     {
21         printf("Мультипликатор: "); scanf("%f",&SecondNumeral); return(Numeral * SecondNumeral);
22     }
23     else if(strcmp(Operation, "/", 1) == 0)
24     {
25         printf("Делитель: ");
26         scanf("%f",&SecondNumeral); if(SecondNumeral == 0)
27         {
28             printf("Ошибка: деление на ноль! "); return(HUGE_VAL);
29         }
30     }
31 }
```

Рис. 3: Готовый файл

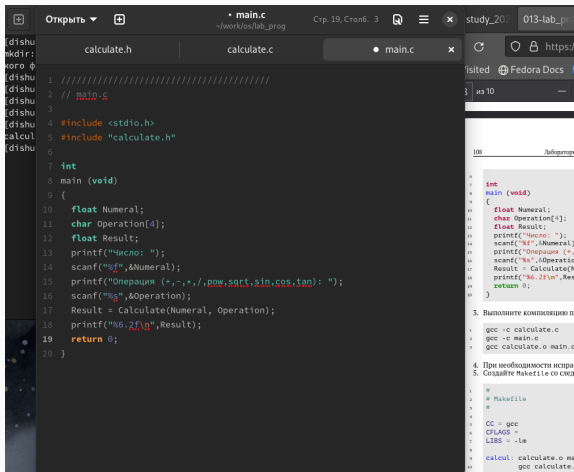
4. Создаем интерфейсный файл `calculate.h`, описывающий формат вызова функции-калькулятора.

A screenshot of a code editor window. The title bar shows the file name 'calculate.h' and the path '~\work\os\lab_prog'. The editor has two tabs: 'calculate.h' (active) and 'calculate.c'. The code in 'calculate.h' is as follows:

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

Рис. 4: Готовый файл

5. Создаем основной файл main.c, реализующий интерфейс пользователя к калькулятору.



The screenshot shows a code editor with a dark theme. The main window displays the content of `main.c`, which includes a header file `calculate.h` and implements the `main` function. The `main` function prompts the user for a number and an operation, then calls the `Calculate` function from `calculate.h` to perform the calculation. A terminal window on the right shows the compilation process using `gcc` and the execution of the resulting `calculate.o` file.

```
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("%g.2f\n",Result);
19     return 0;
20 }
```

3. Выполните компиляцию программы

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

4. При необходимости исправьте ошибки

5. Создайте Makefile со следующим содержанием

```
1 #
2 # Makefile
3 #
4 CC = gcc
5 CFLAGS =
6 LBS = -lm
7
8 calcul: calculate.o main.o
9     gcc calculate.o main.o
```

Рис. 5: Готовый файл

6. Выполняем компиляцию программы посредством gcc.

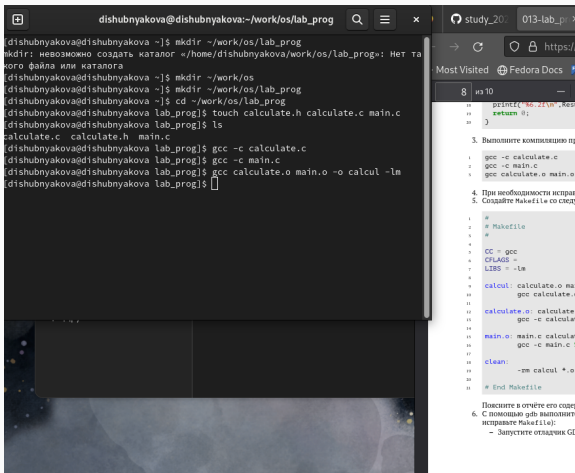
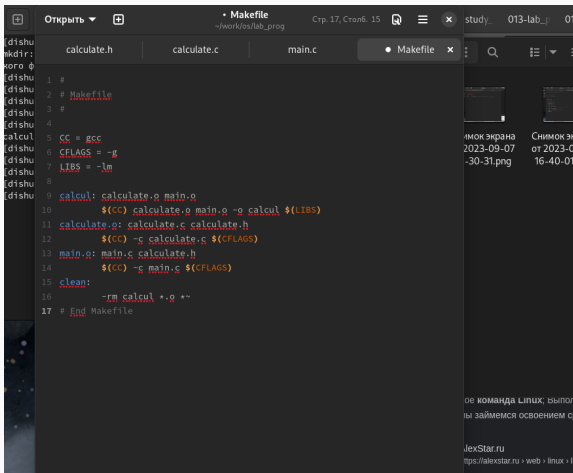


Рис. 6: Команды для компиляции

7. Создаем и редактируем Makefile.



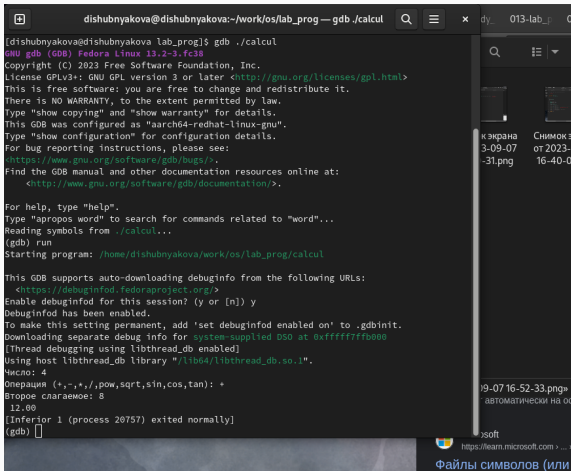
The screenshot shows a code editor with a dark theme. The title bar indicates the file is 'Makefile' located at '~\work\es\lab_prog'. The editor content shows a Makefile with the following rules:

```
1 #
2 # Makefile
3 #
4
5 CC = gcc
6 CFLAGS = -g
7 LIBS = -lm
8
9 calcul: calculate.o main.o
10     $(CC) calculate.o main.o -o calcul $(LIBS)
11 calculate.o: calculate.c calculate.h
12     $(CC) -c calculate.c $(CFLAGS)
13 main.o: main.c calculate.h
14     $(CC) -c main.c $(CFLAGS)
15 clean:
16     -rm calcul *.o *~
17 # End Makefile
```

On the left side of the editor, there is a vertical list of file names: 'calculate.h', 'calculate.c', and 'main.c'. On the right side, there is a sidebar showing a file explorer with two image thumbnails labeled 'Снимок экрана 2023-09-07 16-40-01.png' and 'Снимок экрана 2023-09-07 16-40-01.png'. Below the thumbnails, there is a text snippet: 'е команда Linux; выпол... мы займемся освоением с...'. At the bottom right, there is a URL: 'lexStar.ru' and a link: 'https://alexstar.ru > web > linux > li...'. The status bar at the bottom of the editor shows 'Стр. 17, Стр. 6. 15'.

Рис. 7: Makefile

8. Запускаем отладчик GDB, загрузив в него программу для отладки



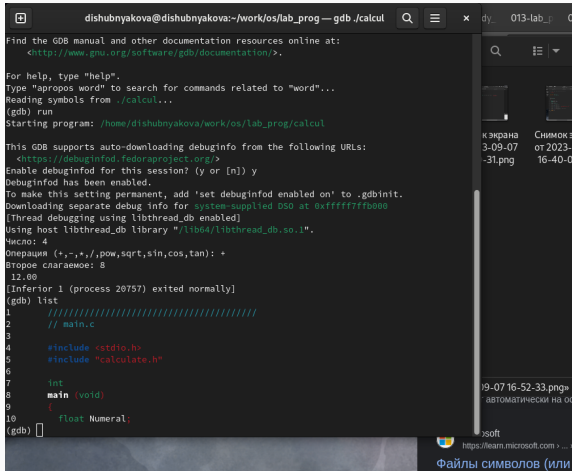
```
dishubnyakova@dishubnyakova:~/work/os/lab_prog — gdb ./calcul
[dishubnyakova@dishubnyakova lab_prog]$ gdb ./calcul
GNU gdb (GDB) Fedora Linux 13.2-3.fc38
Copyright (C) 2023 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: 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 "aarch64-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) run
Starting program: /home/dishubnyakova/work/os/lab_prog/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.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Downloading separate debug info for system-supplied DSO at 0xfffffffffb000
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 4
Операция (+, -, *, /, pow, sqrt, sin, cos, tan): +
Второе слагаемое: 8
12.00
[Inferior 1 (process 20757) exited normally]
(gdb)
```

Рис. 8: Команда gdb ./calcul

9. Запускаем программу и проверяем ее работу.



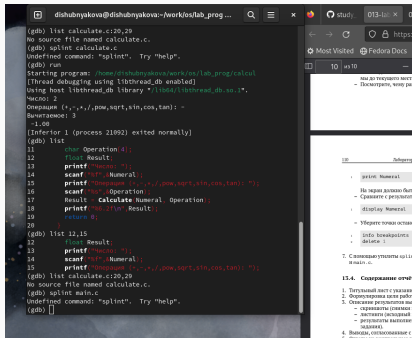
```
dishubnyakova@dishubnyakova:~/work/os/lab_prog — gdb ./calcul
Find the GDB manual and other documentation resources online at:
  <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./calcul...
(gdb) run
Starting program: /home/dishubnyakova/work/os/lab_prog/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.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Downloading separate debug info for system-supplied DSO at 0xfffff7ffb000
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 4
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 8
12.00
[Inferior 1 (process 20757) 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) 
```

Рис. 9: Команда run

10. Используем команду `list` для просмотра определенных строк различных файлов. Смотрим, чему равно на этом этапе значение переменной `Numeral`. Сравниваем с результатом вывода на экран после использования команды. Убираем точки останова и возвращаемся к рабочей программе.



```
(gdb) list calculate.c:20,29
No source file named calculate.c.
(gdb) splint calculate.c
Undefined command: "splint". Try "help".
(gdb) run
Starting program: /home/dishubnyakova/work/oslab_prog/calcul
[thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 2
Операция (+, -, *, /, pow, sqrt, sin, cos, tan): =
Вычисление: 3
-1.00
[Inferior 1 (process 21092) exited normally]
(gdb) list
11      show Operation;
12      float Result;
13      printf("Число: ");
14      scanf("%f", &Numeral);
15      printf("Операция (+, -, *, /, pow, sqrt, sin, cos, tan): ");
16      scanf("%d", &Operation);
17      Result = CalculateNumeral(Numeral, Operation);
18      printf("Результат: %f\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) list calculate.c:20,29
No source file named calculate.c.
(gdb) splint main.c
Undefined command: "splint". Try "help".
(gdb) []
```

The screenshot also shows a web browser window on the right with a URL bar containing 'https://'. Below the browser, there is a sidebar with a list of items, including 'print Numeral', 'info breakpoints', and 'delete 1'. The main content area of the browser shows a list of items, including '1. Титульный лист с указанием', '2. Формулировка задачи работы', '3. Описание результатов работы', '4. Выводы, сделанные в ходе работы', and '5. Ответы на контрольные вопросы'.

Рис. 10: Программа

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

1. Dash P. Getting started with oracle vm virtualbox. Packt Publishing Ltd, 2013. 86 p.
2. Colvin H. Virtualbox: An ultimate guide book on virtualization with virtualbox. CreateSpace Independent Publishing Platform, 2015. 70 p.
3. van Vugt S. Red hat rhcsa/rhce 7 cert guide : Red hat enterprise linux 7 (ex200 and ex300). Pearson IT Certification, 2016. 1008 p.
4. Робачевский А., Немнюгин С., Стесик О. Операционная система unix. 2-е изд. Санкт-Петербург: БХВ-Петербург, 2010. 656 p.
5. Немет Э. et al. Unix и Linux: руководство системного администратора. 4-е изд. Вильямс, 2014. 1312 p.
6. Колисниченко Д.Н. Самоучитель системного администратора Linux. СПб.: БХВ-Петербург, 2011. 544 p.
7. Robbins A. Bash pocket reference. O'Reilly Media, 2016. 156 p.

Спасибо за внимание!