

Лабораторная работа №13. Презентация

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

Козлов Всеволод Павлович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	12
	Список литературы	13

Список иллюстраций

2.1	calculate.c	6
2.2	calculate.h	7
2.3	main.c	7
2.4	Makefile	8
2.5	Запуск программы в отладчике	9
2.6	Команда list	9
2.7	Установка точки останова	10
2.8	Запуск программы	11
2.9	Удаление точки останова	11

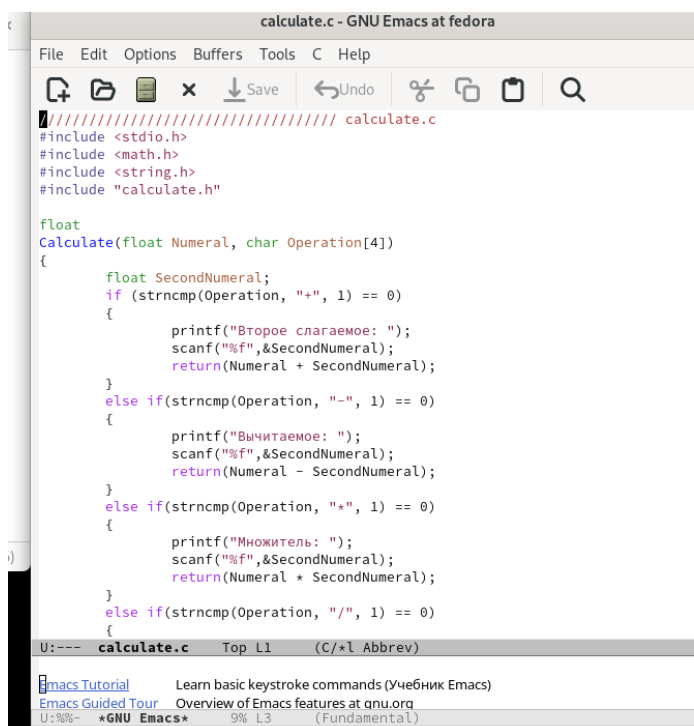
Список таблиц

1 Цель 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.

2 Выполнение лабораторной работы

Написал программу calculate.c (рис. [2.1])



```
calculate.c - GNU Emacs at fedora
File Edit Options Buffers Tools C Help
[Icons: Open, Save, Undo, Cut, Copy, Paste, Find]
// calculate.c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include "calculate.h"

float
Calculate(float Numeral, char Operation[4])
{
    float SecondNumeral;
    if (strcmp(Operation, "+") == 0)
    {
        printf("Второе слагаемое: ");
        scanf("%f", &SecondNumeral);
        return(Numeral + SecondNumeral);
    }
    else if (strcmp(Operation, "-") == 0)
    {
        printf("Вычитаемое: ");
        scanf("%f", &SecondNumeral);
        return(Numeral - SecondNumeral);
    }
    else if (strcmp(Operation, "*") == 0)
    {
        printf("Множитель: ");
        scanf("%f", &SecondNumeral);
        return(Numeral * SecondNumeral);
    }
    else if (strcmp(Operation, "/") == 0)
    {
        printf("Делитель: ");
        scanf("%f", &SecondNumeral);
        return(Numeral / SecondNumeral);
    }
}

U:--- calculate.c Top L1 (C/*l Abbrev)
Emacs Tutorial Learn basic keystroke commands (Учебник Emacs)
Emacs Guided Tour Overview of Emacs features at gnu.org
U:%%- *GNU Emacs* 9% L3 (Fundamental)
```

Рис. 2.1: calculate.c

Написал программу calculate.h (рис. [2.2])

```
calculate.h - GNU Emacs at fedora
File Edit Options Buffers Tools C Help
[Icons: Open, Save, Undo, Redo, Search]
// calculate.h
#ifndef CALCULATE_H_
#define CALCULATE_H_
float Calculate(float Numeral, char Operation[4]);
#endif /*CALCULATE_H_*/
```

Рис. 2.2: calculate.h

Написал программу main.c (рис. [2.3])

```
main.c - GNU Emacs at fedora
File Edit Options Buffers Tools C Help
[Icons: Open, Save, Undo, Redo, Search]
// main.c
#include <stdio.h>
#include "calculate.h"
int main (void)
{
    float Numeral;
    char Operation[4];
    float Result;
    printf("Число: ");
    scanf("%f",&Numeral);
    printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
    scanf("%s",&Operation);
    Result = Calculate(Numeral, Operation);
    printf("%6.2f\n",Result);
    return 0;
}
```

Рис. 2.3: main.c

Написал программу Makefile (рис. [2.4])



The screenshot shows a text editor window titled "Makefile" with the path "~/work/os/lab_prog". The editor contains the following text:

```
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
12 calculate.o: calculate.c calculate.h
13     $(CC) -c calculate.c $(CFLAGS)
14
15 main.o: main.c calculate.h
16     $(CC) -c main.c $(CFLAGS)
17
18 clean:
19     -rm calcul *.o
20
21 # End Makefile
22
```

Рис. 2.4: Makefile

Открыл дебаггер и запустил программу (рис. [2.5])


```
(No debugging symbols found in ./calcul)
(gdb) run
Starting program: /home/vpkozlov/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Число: 7
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 4
11.00
[Inferior 1 (process 2827) exited normally]
(gdb) █
```

Рис. 2.5: Запуск программы в отладчике

Просмотрел строки кода с помощью команды list (рис. [2.6])

```
[Inferior 1 (process 3301) exited normally]
(gdb) list
1  //////////////////////////////////////
2  // main.c
3
4  #include <stdio.h>
5  #include "calculate.h"
6
7  int main (void)
8  {
9      float Numeral;
10     char Operation[4];
(gdb) list 12, 15
12     printf("Число: ");
13     scanf("%f",&Numeral);
14     printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
15     scanf("%s",&Operation);
(gdb) █
```

Рис. 2.6: Команда list

Установил точку останова (рис. [2.7])

```
(gdb) break 21
Breakpoint 1 at 0x401234: file calculate.c, line 21.
(gdb) info breakpoints
Num    Type             Disp Enb Address            What
1      breakpoint      keep y   0x0000000000401234 in calculate
                                at calculate.c:21
(gdb)
```

Рис. 2.7: Установка точки останова

Запуск программы с точкой останова (рис. [2.8])

```
vpkozlov@fedora:~/work/os/lab_prog — gdb ./calcul
31     printf("Делитель: ");
16     }
17     else if(strncmp(Operation, "-", 1) == 0)
18     {
19         printf("Вычитаемое: ");   ение на ноль! ");
20         scanf("%f",&SecondNumeral);
21         return(Numeral - SecondNumeral);
22     }
23     else if(strncmp(Operation, "+", 1) == 0)
24     {
25         return(Numeral / SecondNumeral);
26     }
27     printf("Множитель: ");
28     scanf("%f",&SecondNumeral);
29     return(Numeral * SecondNumeral);
??    PC: ??
Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffffdf34 "-") at calculate.c:21
(gdb) print Numeral
$1 = 5
(gdb)
```

Рис. 2.8: Запуск программы

Удаление точки останова (рис. [2.9])

```
multi-thre Thread 0x7ffff7ecd7 In: Calculate at calculate.c:21 PC: 0x401234
1: Numeral = 5
(gdb) info breakpoints
Num Type Disp Enb Address What
1 breakpoint keep y 0x0000000000401234 in Calculate
at calculate.c:21
breakpoint already hit 1 time
(gdb) delete 1
(gdb)
```

Рис. 2.9: Удаление точки останова

3 Выводы

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.

Список литературы