

# Презентация по лабораторной работе № 13

---

Королёв И.А

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Королёв Иван Андреевич
- студент, НКАбд - 05 - 22
- Российский университет дружбы народов

## 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.

## Задание

---

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

## Теоретическое введение

---




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;

## Выполнение лабораторной работы

---

# Подготовка файлов для создания примитивного калькулятора

```
Открыть ▼   
14 |  
15 printf("Второе слагаемое: ");  
16 scanf("%f",&SecondNumeral);  
17 return(Numeral + SecondNumeral);  
18 |  
19 else if(strncmp(Operation, "-", 1) == 0)  
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);  
30 }  
31 else if(strncmp(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(strncmp(Operation, "pow", 3) == 0)  
44 {  
45 printf("Степень: ");  
46 scanf("%f",&SecondNumeral);  
47 return(pow(Numeral, SecondNumeral));  
48 }  
49 else if(strncmp(Operation, "sqrt", 4) == 0)  
50 return(sqrt(Numeral));  
51 else if(strncmp(Operation, "sin", 3) == 0)  
52 return(sin(Numeral));  
53 else if(strncmp(Operation, "cos", 3) == 0)  
54 return(cos(Numeral));  
55 else if(strncmp(Operation, "tan", 3) == 0)  
56 return(tan(Numeral));  
57 else  
58 {  
59 printf("Неправильно введено действие ");  
60 return(HUGE_VAL);
```

## Выполните компиляцию

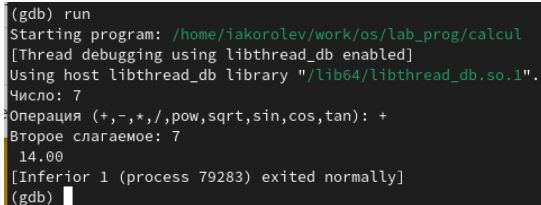
```
[iakorolev@fedora lab_prog]$ gcc -c main.c  
[iakorolev@fedora lab_prog]$ gcc -c calculate.c  
[iakorolev@fedora lab_prog]$ gcc calculate.o main.o -o calcul -lm  
[iakorolev@fedora lab_prog]$
```

Рис. 2: Выполните компиляцию

## Создайте Makefile

```
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
22
```

Для запуска программы внутри отладчика введите команду run (рис. (fig:007?)).



```
(gdb) run
Starting program: /home/iakorolev/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: 7
14.00
[Inferior 1 (process 79283) exited normally]
(gdb) 
```

Рис. 4: run

Для постраничного (по 9 строк) просмотра исходного код используйте команду list (рис. (fig:008?)).

```
[Inferior 1 (process 80553) exited normally]
(gdb) list
1      //////////////////////////////////////
2      // main.c
3
4      #include <stdio.h>
5      #include "calculate.h"
6      int main (void)
7      {
8          float Numeral;
9          char Operation[4];
10         float Result;
(gdb) list
11         printf("Число: ");
12         scanf("%f",&Numeral);
13         printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
14         scanf("%s",Operation);
15         Result = Calculate(Numeral, Operation);
16         printf("%6.2f\n",Result);
17         return 0;
18     }
(gdb) █
```

Рис. 5: list

Для просмотра определённых строк не основного файла используйте `list` с параметрами (рис. (fig:0010?)).

```
19     result = calculate(Numeral1, operation);  
(gdb) list calculate.c:20,29  
20     {  
21         printf("Вычитаемое: ");  
22         scanf("%f",&SecondNumeral);  
23         return(Numeral1 - SecondNumeral);  
24     }  
25     else if(strncmp(Operation, "*", 1) == 0)  
26     {  
27         printf("Множитель: ");  
28         scanf("%f",&SecondNumeral);  
29         return(Numeral1 * SecondNumeral);  
(gdb) █
```

Рис. 6: list calculate.c:20,29



Установите точку остановки в файле calculate.c на строке номер 21 (рис. (fig:0011?)).

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

Рис. 7: break 21

Запустите программу внутри отладчика и убедитесь, что программа остановится в момент прохождения точки остановки.

```
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): -

Breakpoint 1, Calculate (Numeral=5, Operation=0x7fffffffdee4 "-")
    at calculate.c:21
21      printf("Вычитаемое: ");
(gdb) print Numeral
$1 = 5
(gdb) display Numeral
1: Numeral = 5
(gdb) delete 1
(gdb) info breakpoints
No breakpoints or watchpoints.
(gdb) █
```

Рис. 8: run

С помощью утилиты splint анализирую кода файла main.c (рис. (fig:0015?)).

```
[iakorolev@fedora lab_prog1]$ 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:12:5: 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:14:5: Return value (type int) ignored: scanf("%s", Oper...

Finished checking --- 3 code warnings
[iakorolev@fedora lab_prog1]$
```

Рис. 9: splint

## Выводы

---

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