Лабораторная работа 13

ПОДГОТОВИЛ

МАНАТОВ РАМАЗАН

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.

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.

```c
#include <stdio.h>
#include <math.h>
#include <string.h>
#include "calculate.h"

float
Calculate (float Numeral, char Operation[4])
{
  float SecondNumeral;
  if(strncmp(Operation, "+", 1) == 0)
    {
      printf("Второе слагаемое ");
      scanf("%f",&SecondNumeral);
      return(Numeral + SecondNumeral);
    }
  else if(strncmp(Operation, "-", 1) == 0)
    {
      printf("Вычитаемое: ");
      scanf("%f",&SecondNumeral);
      return(Numeral - SecondNumeral);
    }
  else if(strncmp(Operation, "*", 1) == 0)
    {
      printf("Множитель: ");
      scanf("%f",&SecondNumeral);
      return(Numeral * SecondNumeral);
    }
  else if(strncmp(Operation, "/", 1) == 0)
    {
      printf("Делитель: ");
      scanf("%f",&SecondNumeral);
      if(SecondNumeral == 0)
        {
          printf("Ошибка: деление на ноль! ");
          return(HUGE_VAL);
        }
      else
        return(Numeral / SecondNumeral);
    }
  else if(strncmp(Operation, "pow", 3) == 0)
    {
      printf("Степень: ");
      scanf("%f",&SecondNumeral);
      printf("Степень: ");
      scanf("%f",&SecondNumeral);
      return(pow(Numeral, SecondNumeral));
    }
  else if(strncmp(Operation, "sqrt", 4) == 0)
    return(sqrt(Numeral));
  else if(strncmp(Operation, "sin",3) == 0)
    return(sin(Numeral));
  else if(strncmp(Operation, "cos", 3) == 0)
    return(cos(Numeral));
  else if(strncmp(Operation, "tan", 3) == 0)
    return(tan(Numeral));
  else
    {
      printf("Неправильно введено действие");
      return(HUGE_VAL);
    }
}
```

Интерфейсный файл calculate.h, описывающий формат вызова функции калькулятора

```c
//calculate.h

#ifndef CALCULATE_H_
#define CALCULATE_H_

float Calculate(float Numeral, char Operation[4]);

#endif /*CALCULATE_H_*/
```

Основной файл main.c, реализующий интерфейс пользователя к калькулятору

```c
//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;
}
```

Создал Makefile с необходимым содержанием

```
#
# Makefile
#

 CC = gcc
 CFLAGS =
 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)

 clean:
         -rm calcul *.o *~

# End Makefile#
```

Далее исправил Makefile

```
#
# Makefile
#

 CC = gcc
 CFLAGS = -g
 LIBS = -lm

 calcul: calculate.o main.o
         $(CC) calculate.o main.o -o calcul $(LIBS)

 calculate.o: calculate.c calculate.h
         $(CC) -c calculate.c $(CFLAGS)

 main.o: main.c calculate.h
         $(CC) -c main.c $(CFLAGS)

 clean:
         -rm calcul *.o *~

# End Makefile#
```

В переменную CFLAGS добавил опцию -g, необходимую для компиляции объектных файлов и их использования в программе отладчика GDB. Сделала так, что утилита компиляции выбирается с помощью переменной CC. После этого я удалил исполняемые и объектные файлы из каталога с помощью команды

Запуск программы

```
(gdb) run
Starting program: /home/rrManatov/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): +
Второе слагаемое 1
6.00
%6,2f/n[Inferior 1 (process 4256) exited normally]
(gdb)
```

Команды gdb

Для постраничного (по 10 строк) просмотра исходного кода использовал команду «list»

```
(gdb) list
16        Result = Calculate(Numeral, Operation);
17        printf("%6,2f/n",Result);
18        return 0;
19    }
```

Для просмотра строк с 12 по 15 основного файла использовал команду «list 12,15»

```
(gdb) list 12,15
12        printf("Число: ");
13        scanf("%f",&Numeral);
14        printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan)
: ");
15        scanf("%s",&Operation);
```

```
(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          }
```

информацию об имеющихся в проекте точках останова с помощью команды «info breakpoints

```
(gdb) info breakpoints
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x000000000040121e in Cal
culate at calculate.c:21
```

ВЫВОД

В ходе выполнения данной 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