Отчёт по лабораторной работе №13

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

Кочина Д. С.

03 мая 2023

Российский университет дружбы народов, Москва, Россия

Вводная часть

Цель работы

Целью данной лабораторной работы является приобретение простейших навыков разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования C калькулятора с простейшими функциями.

Основная часть

Программа в calculate.c

- Написала программу - примитивнейший калькулятор, способный складывать, вычитать, умножать и делить, возводить число в степень, брать квадратный корень, вычислять sin, cos, tan. При запуске он запрашивал первое число, операцию, второе число. После этого программа выводила результат и останавливалась.

Программа в calculate.c

```
/////////////////////////////////////////////////
// calculate.c

#include<studio.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);
```

Программа в calculate.c

```
    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)
```

Программа в calculate.c

```
  else if(strncmp(Operation, "pow" , 3) == 0)
    {
      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);
    }
}
```

Программа в calculate.h

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

```
/// /// /// /// /// /// /// /// /// /// /// /// /// /// /// /// /// /// /// /// ///
// calculate.h

#ifndef CALCULATE_H_
#define CALCULATE_H_

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

#endif /*CALCULATE_H_*/
```

Программа в main.c

- Основной файл main.c, реализующий интерфейс пользователя к калькулятору.

```
/////////////////////////////////////////////////////////////
// main.c

#include <studio.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);
```

Компиляция программы

- Выполнила компиляцию программы посредством gcc, используя команды «gcc -c calculate.c», «gcc -c main.c» и «gcc calculate.o main.o -o calcul -lm».
- В ходе компиляции программы никаких ошибок выявлено не было.

```
dskochina@dk3n31 ~/work/os/lab_prog $ gcc -c calculate.c
dskochina@dk3n31 ~/work/os/lab_prog $ gcc -c main.c
dskochina@dk3n31 ~/work/os/lab_prog $ gcc calculate.o main.o -o calcul -lm
dskochina@dk3n31 ~/work/os/lab_prog $
```

Программа в Makefile

- Создала Makefile с необходимым содержанием.
- Далее исправила Makefile.

Программа в 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
```

Удаление и компиляция файлов

```
dskochina@dk3n31 ~/work/os/lab_prog $ make clean
rm calcul *.o *~
dskochina@dk3n31 ~/work/os/lab_prog $ make calculate.o
gcc -c calculate.c -g
dskochina@dk3n31 ~/work/os/lab_prog $ make main.o
gcc -c main.c -g
dskochina@dk3n31 ~/work/os/lab_prog $ make calcul
gcc calculate.o main.o -o calcul -lm
dskochina@dk3n31 ~/work/os/lab_prog $
```

Работа с gdb

- С помощью gdb выполнила отладку программы calcul. Запустила отладчик GDB, загрузив в него программу для отладки, используя команду: «gdb./calcul».
- Для запуска программы внутри отладчика ввела команду «run».
- Для постраничного просмотра исходного кода использовала команду «list».
- Для просмотра строк с 12 по 15 основного файла использовала команду «list 12,15».
- Для просмотра определённых строк не основного файла использовала команду «list calculate.c:20,29».
- Установила точку останова в файле calculate.c на строке номер 21, используя команды «list calculate.c:20,27» и «break 21».
- Вывела информацию об имеющихся в проекте точках останова с помощью команды «info breakpoints».
- Запустила программу внутри отладчика и убедилась, что программа остановилась в момент прохождения точки останова. Использовала команды «run», «5», «*» и «backtrace».

Работа с gdb

```
dskochina@dk3n31 ~/work/os/lab_prog $ gdb ./calcul
GNU gdb (Gentoo 12.1 vanilla) 12.1
Copyright (C) 2022 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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://bugs.gentoo.org/>.
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...
```

Работа с gdb

```
(gdb) info breakpoints
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000555555555247 in Calculate at calculate.c:21
        breakpoint already hit 1 time
(gdb) delete 1
(gdb)
```

Команда splint calculate.c

Воспользовалась командами «splint calculate.c» и «splint main.c».

```
dskochina@dk3n31 ~/work/os/lab_prog $ splint calculate.c
Splint 3.1.2 --- 07 Dec 2021

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)
calculate.c:10:31: Function parameter Operation declared as manifest array
                     (size constant is meaningless)
calculate.c: (in function Calculate)
calculate.c:16:1: Return value (type int) ignored: scanf("%f", &Sec...
  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)
calculate.c:22:1: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:1: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:1: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:4: Dangerous equality comparison involving float types:
                    SecondNumeral == 0
  Two real (float, double, or long double) values are compared directly using
  == or != primitive. This may produce unexpected results since floating point
  representations are inexact. Instead, compare the difference to FLT_EPSILON
  or DBL_EPSILON. (Use -realcompare to inhibit warning)
calculate.c:38:7: Return value type double does not match declared type float:
                    (HUGE_VAL)
```

Команда splint main.c

```
dskochina@dk3n31 ~/work/os/lab_prog $ splint main.c
Splint 3.1.2 --- 07 Dec 2021

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:14:3: 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:16:3: Return value (type int) ignored: scanf("%s", Oper...

Finished checking --- 3 code warnings
dskochina@dk3n31 ~/work/os/lab_prog $
```

Заключение

Вывод

В ходе выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX. А также приобрела практические навыки написания более сложные командных файлов с использованием логических управляющих конструкций и циклов.