

Отчёт по лабораторной работе №11

Средства, применяемые при разработке программного обеспечения в ОС типа Linux

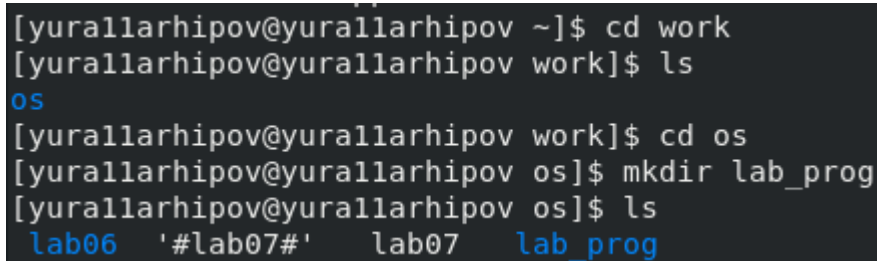
Архипов Юрий Денисович

Цель работы

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

Ход работы

1. В домашнем каталоге создать подкаталог ~/work/os/lab_prog.



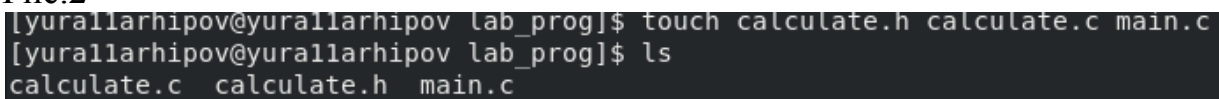
```
[yurallarhipov@yurallarhipov ~]$ cd work
[yurallarhipov@yurallarhipov work]$ ls
os
[yurallarhipov@yurallarhipov work]$ cd os
[yurallarhipov@yurallarhipov os]$ mkdir lab_prog
[yurallarhipov@yurallarhipov os]$ ls
lab06  '#lab07#'  lab07  lab_prog
```

Рис.1

2. Создать в подкаталоге файлы: calculate.h, calculate.c, main.c. Написать примитивный калькулятор, способный складывать, вычитать, умножать и делить, возводить число в степень, брать квадратный корень, вычислять sin, cos и tan. При запуске он будет запрашивать первое число, операцию, второе число. После этого программа выведет результат и остановится.

а) Создадим файлы в подкаталоге.

Рис.2



```
[yurallarhipov@yurallarhipov lab_prog]$ touch calculate.h calculate.c main.c
[yurallarhipov@yurallarhipov lab_prog]$ ls
calculate.c  calculate.h  main.c
```

б) Пишем скрипт.

```
#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);
    }if(strncmp(Operation, "*", 1) == 0){
        printf("Множитель: ");
        scanf("%f",&SecondNumeral);
        return(Numeral * SecondNumeral);
    }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);
        return(Numeral + SecondNumeral);
    }
    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);
    }
}
```

Рис.3

Рис.4

Рис.5

```

#ifndef CALCULATE_H_
#define CALCULATE_H_
float Calculate(float Numeral, char Operation[4]);
#endif /*CALCULATE_H_*/

```

Рис.6

```

#include <stdio.h>
#include "calculate.h"

main(void){

    float Numeral;
    char Operation[4];
    float Result;
    printf("Num: \n");
    scanf("%f",&Numeral);
    printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): \n ");
    scanf("%s",Operation);
    Result = Calculate(Numeral,Operation);
    printf("%6.2f\n",Result);
    return 0;
}

```

3-4. Выполнить компиляцию программы посредством gcc, при необходимости исправить синтаксические ошибки.

а) Выполняем компиляцию программы, исправив ошибки.

Рис.7 [yurallarhipov@yurallarhipov lab_prog]\$ gcc -c calculate.c

Рис.8 [yurallarhipov@yurallarhipov lab_prog]\$ gcc -c main.c

Рис.9

[yurallarhipov@yurallarhipov lab_prog]\$ gcc calculate.o main.o -o calcul -lm

5. Создать Makefile с содержанием из лабораторной работы.

а) Создаём файл.

Рис.10 [yurallarhipov@yurallarhipov lab_prog]\$ touch Makefile
[yurallarhipov@yurallarhipov lab_prog]\$ emacs Makefile

б) Пишем программу.

```
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 *~
```

Рис.11

6. С помощью gdb выполнить отладку программы calcul.

а) Запускаем отладчик GDB, загрузив в него программу для отладки.

Рис.12

```
[yurallarhipov@yurallarhipov lab_prog]$ gdb ./calcul
GNU gdb (GDB) Red Hat Enterprise Linux 8.2-16.el8
Copyright (C) 2018 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-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://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...(no debugging symbols found)...done.
```

б) Вводим команду run для запуска программы внутри отладчика.

Рис.13

```
(gdb) run
Starting program: /home/yurallarhipov/work/os/lab_prog/calcul
Missing separate debuginfos, use: yum debuginfo-install glibc-2.28-164.el8.x86_64
4
Num:
4
Операция (+,-,*,/,pow,sqrt,sin,cos,tan):
sqrt
2.00
[Inferior 1 (process 26963) exited normally]
```

с) Команда list и break.

Рис.14

```
(gdb) list
No symbol table is loaded.  Use the "file" command.
(gdb) info breakpoints
```

Рис.15

```
(gdb) list 12,15
No symbol table is loaded.  Use the "file" command.
```

Рис.16

```
(gdb) list calculate.c:20,29
No symbol table is loaded.  Use the "file" command.
```

Рис.17

```
(gdb) break 21
No symbol table is loaded.  Use the "file" command.
```

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

Рис.18

```
[root@yurallarhipov lab_prog]# splint calculate.c
bash: splint: команда не найдена...
```

Рис.19

```
[root@yurallarhipov lab_prog]# yum install splint
Последняя проверка окончания срока действия метаданных: 1:59:49 назад, Вт 23 ноя 2021 06:35:51.
Нет соответствия аргументу: splint
Ошибка: Совпадений не найдено: splint
```

Вывод

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