презентация по лабораторной работе 13

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Боровиков Д.А.

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Боровиков Даниил Александрович
- Студент ФМиЕН РУДН
- Группа НПИбд-01-22

Вводная часть

Цели и задачи

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

Создание нового подкаталога и файлов в нем

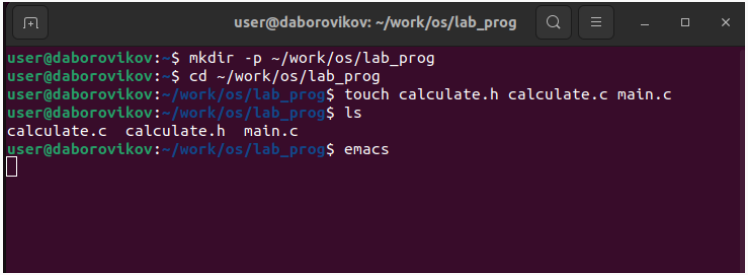


Рис. 1: Создание нового подкаталога и файлов в нем

Реализация функций калькулятора в файле calculate.h:

emacs@daborovikov

File Edit Options Buffers Tools C Help

Save Undo

```
////////////////////////////////////
// calculate.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);
```

U:--- **calculate.c** Top L34 (C/*l Abbrev)

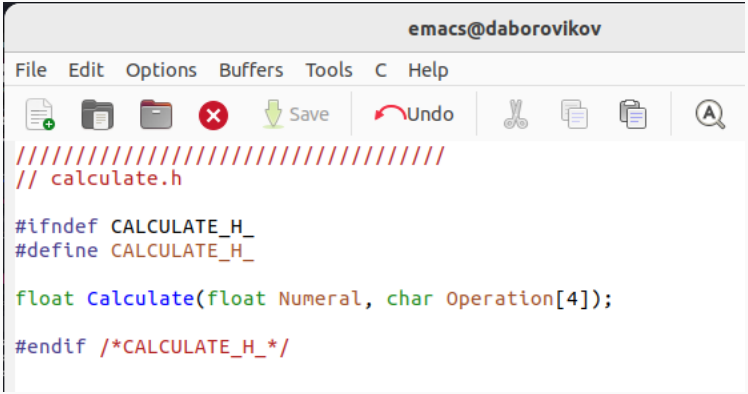


Рис. 3: Интерфейсный файл calculate.h, описывающий формат вызова функции калькулятора:

Основной файл main.c, реализующий интерфейс пользователя к калькулятору

emacs@daborovikov

File Edit Options Buffers Tools C Help

Save Undo

```
////////////////////////////////////
// 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;
}
```

Компиляция программы посредством gcc:

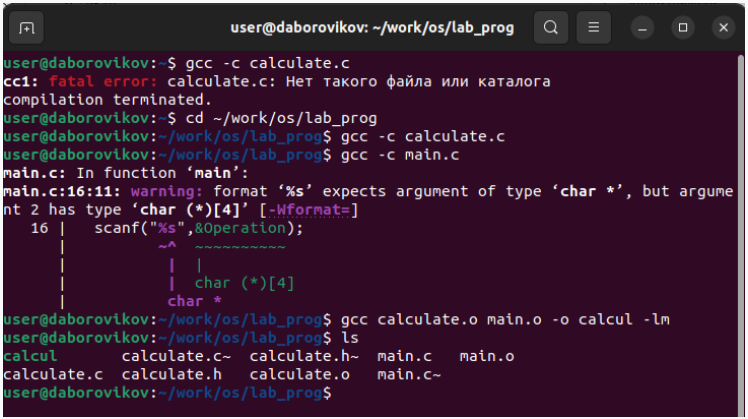
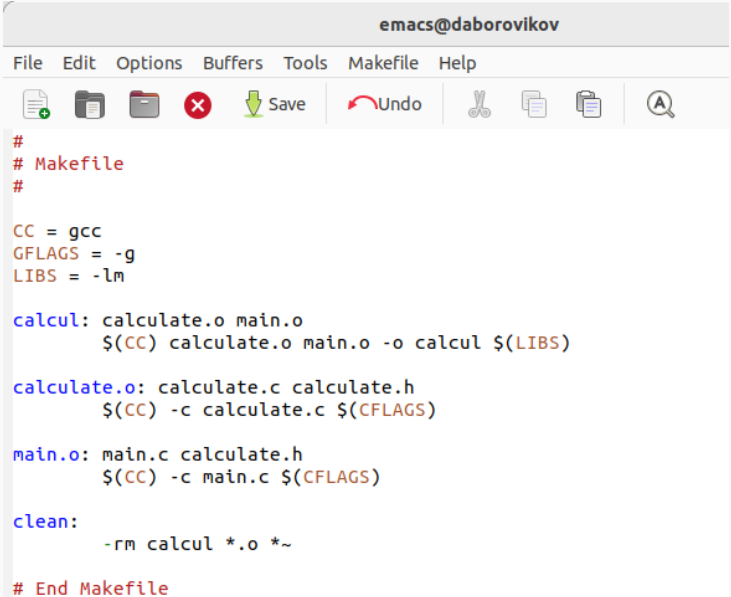


Рис. 5: Компиляция программы посредством gcc:

Создайте Makefile со следующим содержанием:




```
#
# Makefile
#

CC = gcc
GFLAGS = -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
```

Отладка программы calcul

```
user@daborovikov:~/work/os/lab_prog$ emacs Makefile
user@daborovikov:~/work/os/lab_prog$ emacs
user@daborovikov:~/work/os/lab_prog$ make clean
rm calcul *.o *~
user@daborovikov:~/work/os/lab_prog$ make calculate.o
gcc -c calculate.c
user@daborovikov:~/work/os/lab_prog$ make main.o
gcc -c main.c
main.c: In function ‘main’:
main.c:16:11: warning: format ‘%s’ expects argument of type ‘char *’, but argument 2 has type ‘char (*)[4]’ [-Wformat=]
   16 |   scanf("%s",&Operation);
      |          ~^   ~~~~~~~~~
      |           |   |
      |           |   char (*)[4]
      |           char *
user@daborovikov:~/work/os/lab_prog$ make calcul
gcc calculate.o main.o -o calcul -lm
user@daborovikov:~/work/os/lab_prog$ gdb ./calcul
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
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-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://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 in ./calcul)
(gdb) run
Starting program: /home/user/work/os/lab_prog/calcul
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Число: 5
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): *
Множитель: 6
 30.00
[Inferior 1 (process 30033) exited normally]
(gdb)
```

Анализ кода файла calculate.

```
user@daborovikov:~/work/os/lab_prog$ splint calculate.c
Splint 3.1.2 --- 21 Feb 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:16:7: 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:22:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:10: 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:38:10: Return value type double does not match declared type float:
                     (HUGE_VAL)
  To allow all numeric types to match, use +relaxtypes.
calculate.c:46:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:47:13: Return value type double does not match declared type float:
                     (pow(Numeral, SecondNumeral))
calculate.c:50:11: Return value type double does not match declared type float:
                     (sqrt(Numeral))
calculate.c:52:11: Return value type double does not match declared type float:
                     (sin(Numeral))
calculate.c:54:11: Return value type double does not match declared type float:
                     (cos(Numeral))
calculate.c:56:11: Return value type double does not match declared type float:
                     (tan(Numeral))
calculate.c:60:13: Return value type double does not match declared type float:
                     (HUGE_VAL)

Finished checking --- 15 code warnings
user@daborovikov:~/work/os/lab_prog$
```

Анализ кода файла main.c.

```
user@daborovikov:~/work/os/lab_prog$ splint calculate.c
Splint 3.1.2 --- 21 Feb 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:16:7: 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:22:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:28:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:10: 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:38:10: Return value type double does not match declared type float:
                     (HUGE_VAL)
  To allow all numeric types to match, use +relaxtypes.
calculate.c:46:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:47:13: Return value type double does not match declared type float:
                     (pow(Numeral, SecondNumeral))
calculate.c:50:11: Return value type double does not match declared type float:
                     (sqrt(Numeral))
calculate.c:52:11: Return value type double does not match declared type float:
                     (sin(Numeral))
calculate.c:54:11: Return value type double does not match declared type float:
                     (cos(Numeral))
calculate.c:56:11: Return value type double does not match declared type float:
                     (tan(Numeral))
calculate.c:60:13: Return value type double does not match declared type float:
                     (HUGE_VAL)

Finished checking --- 15 code warnings
user@daborovikov:~/work/os/lab_prog$
```

Вывод

В ходе лабораторной работы мы прибрели простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования C калькулятора с простейшими функциями