

# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

## Факультет физико-математических и естественных наук

### ОТЧЕТ

#### ПО ЛАБОРАТОРНОЙ РАБОТЕ №13

дисциплина: *Операционные системы*

Студент: Мохаммад Амин Шоаибуллах Группа: НПИбд-02-20

### МОСКВА

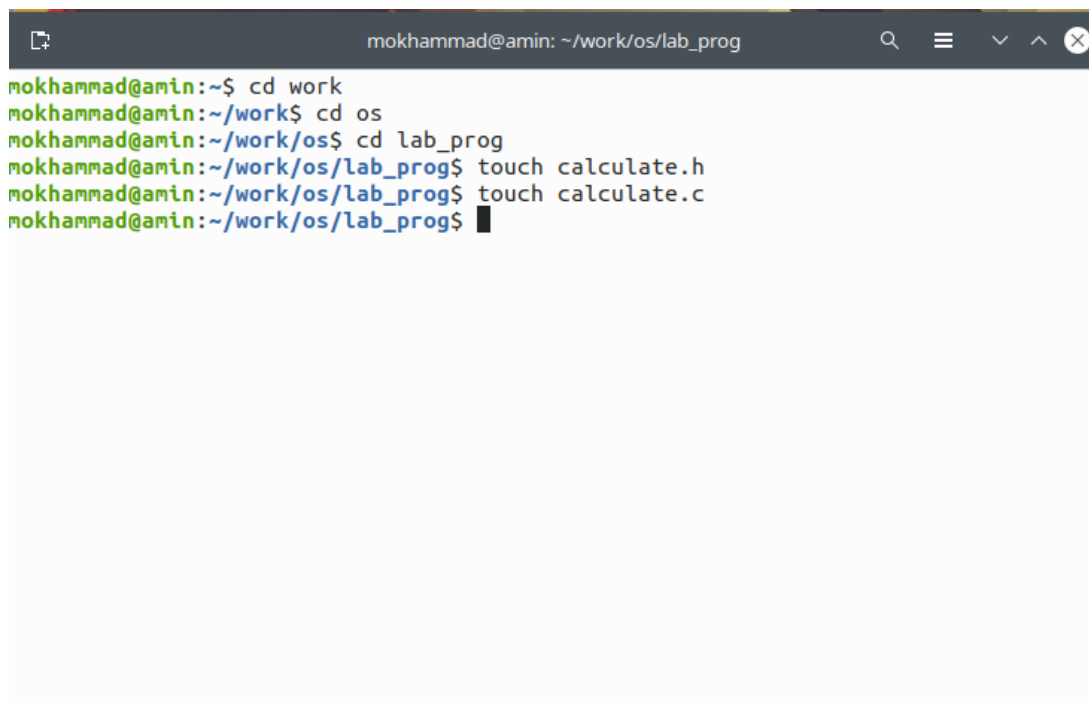
2021 г

#### 1. Цель работы:

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

2. Описание процесса выполнения задания.

- В домашнем каталоге создан подкаталог ~/work/os/lab\_prog. - Созданы файлы: calculate.h, calculate.c, main.c. (см. рис. 1-4).












```
mokhammad@amin: ~/work/os/lab_prog

mokhammad@amin:~$ cd work
mokhammad@amin:~/work$ cd os
mokhammad@amin:~/work/os$ cd lab_prog
mokhammad@amin:~/work/os/lab_prog$ touch calculate.h
mokhammad@amin:~/work/os/lab_prog$ touch calculate.c
mokhammad@amin:~/work/os/lab_prog$
```

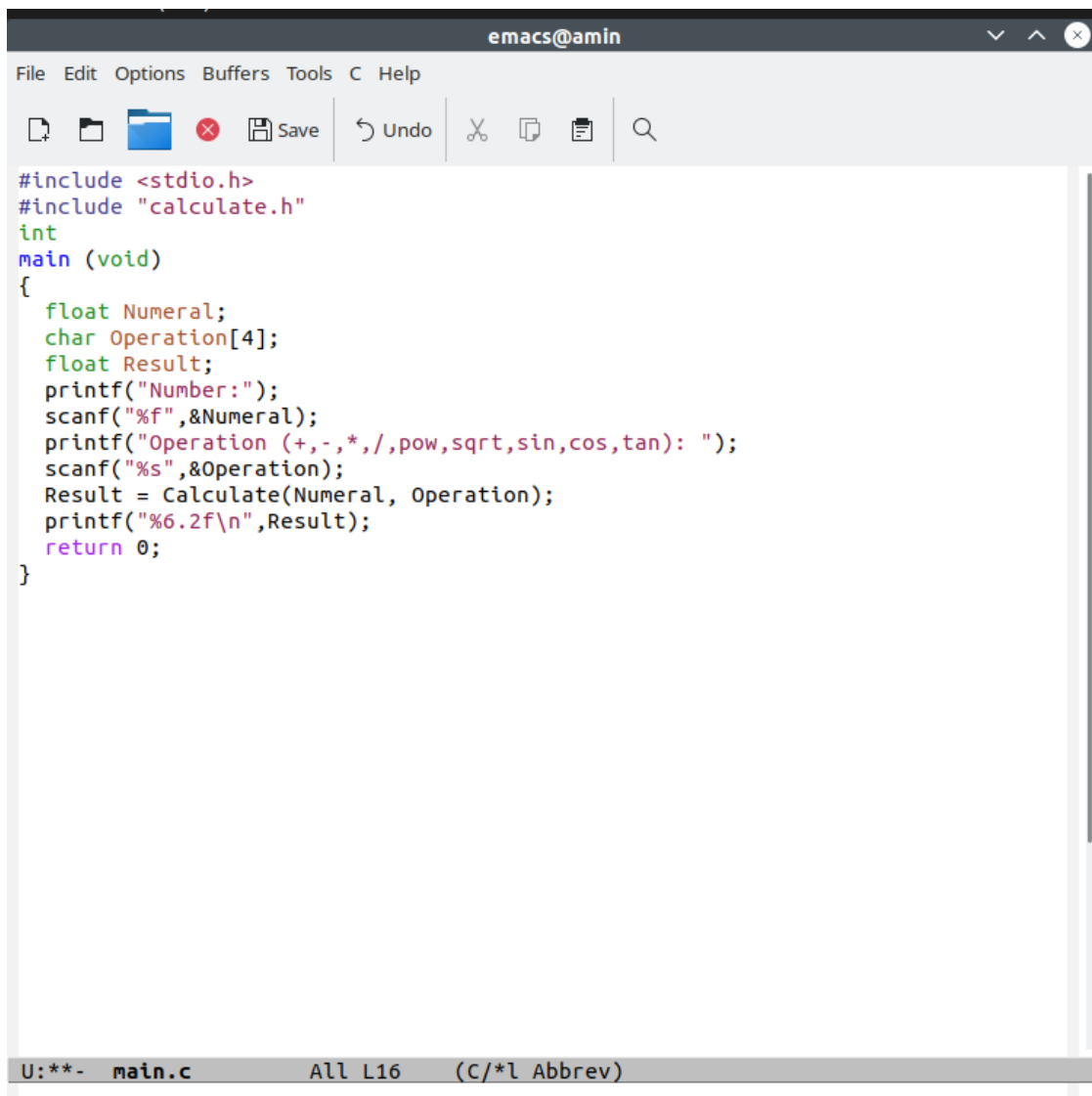
emacs@amin

File Edit Options Buffers Tools C Help

 Save |  Undo |  | 

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

U:\*\*- calculate.h All L5 (C/\*l Abbrev)



The image shows a screenshot of an Emacs editor window. The title bar at the top reads "emacs@amin". Below the title bar is a menu bar with the following items: "File", "Edit", "Options", "Buffers", "Tools", "C", and "Help". Under the "File" menu, there is a toolbar with icons for opening a file, saving a file, undo, redo, cut, copy, paste, and search. The main editing area contains the following C code:

```
#include <stdio.h>
#include "calculate.h"
int
main (void)
{
    float Numeral;
    char Operation[4];
    float Result;
    printf("Number:");
    scanf("%f",&Numeral);
    printf("Operation (+,-,*,/,pow,sqrt,sin,cos,tan): ");
    scanf("%s",&Operation);
    Result = Calculate(Numeral, Operation);
    printf("%6.2f\n",Result);
    return 0;
}
```

At the bottom of the window, there is a status bar that displays "U:\*\*- main.c", "All L16", and "(C/\*l Abbrev)".

emacs@amin

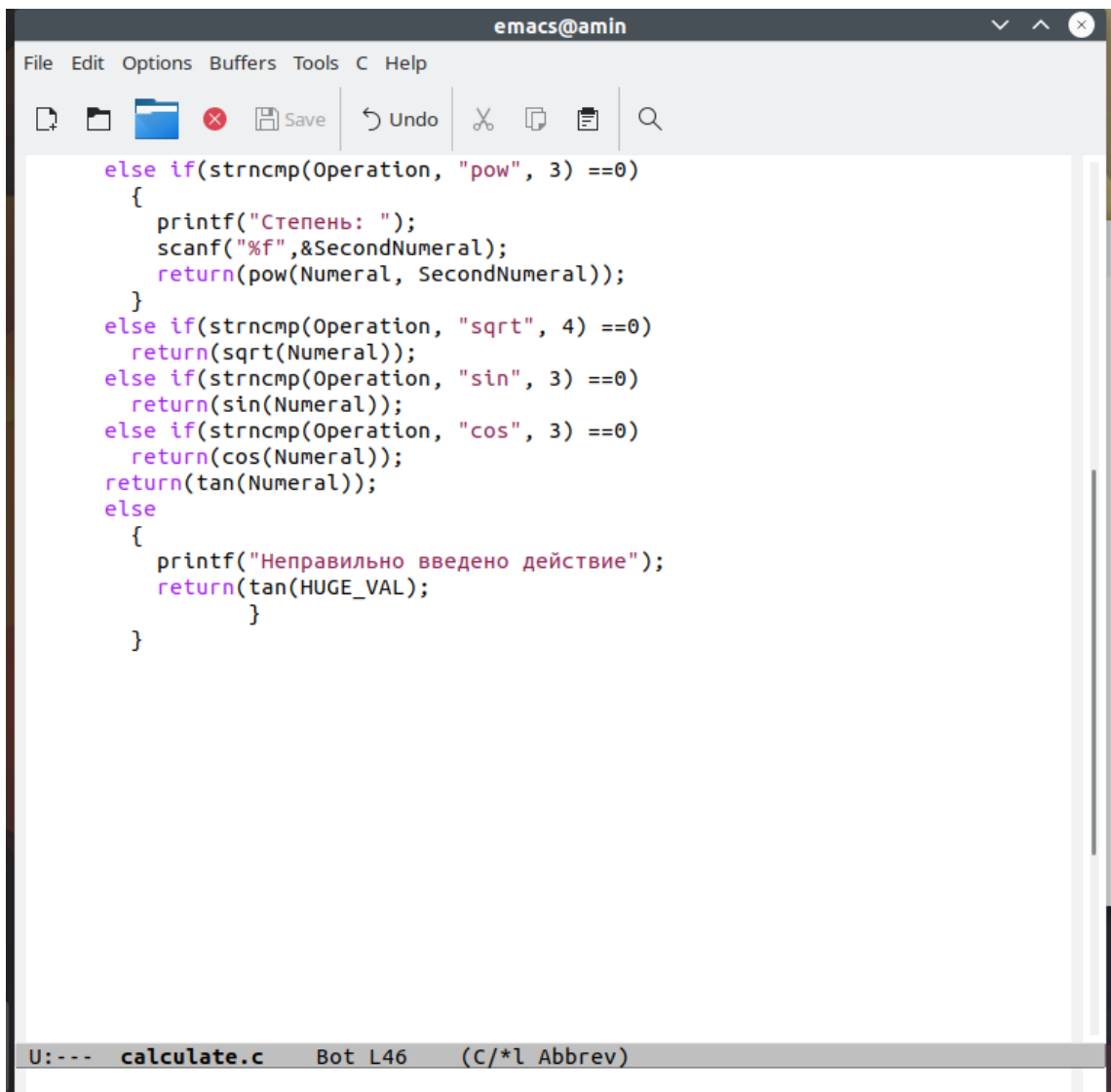
File Edit Options Buffers Tools C Help

Save Undo

```
#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);
        }
    }
}
```

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

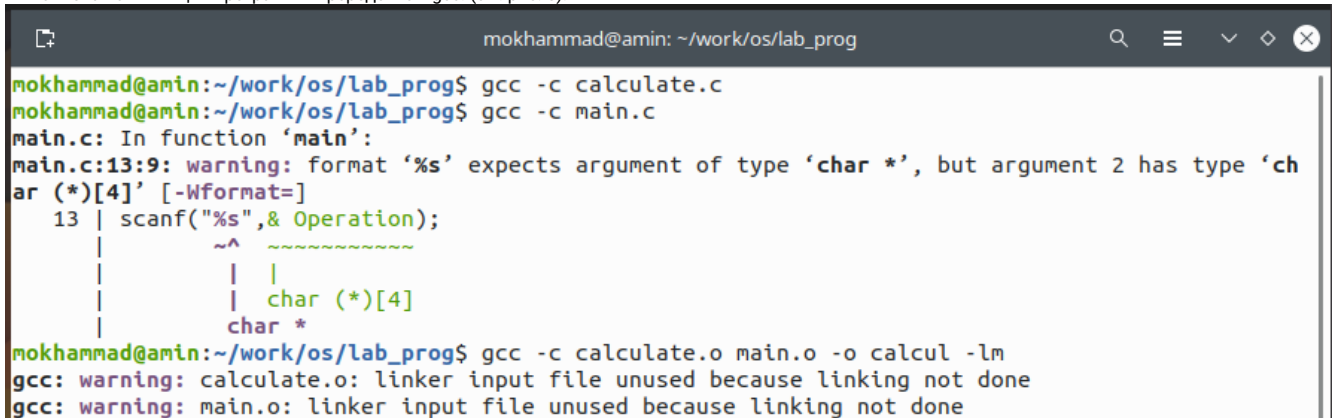
Beginning of buffer



```
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));
return(tan(Numeral));
else
{
    printf("Неправильно введено действие");
    return(tan(HUGE_VAL));
}
}
```

U: --- calculate.c Bot L46 (C/\*l Abbrev)

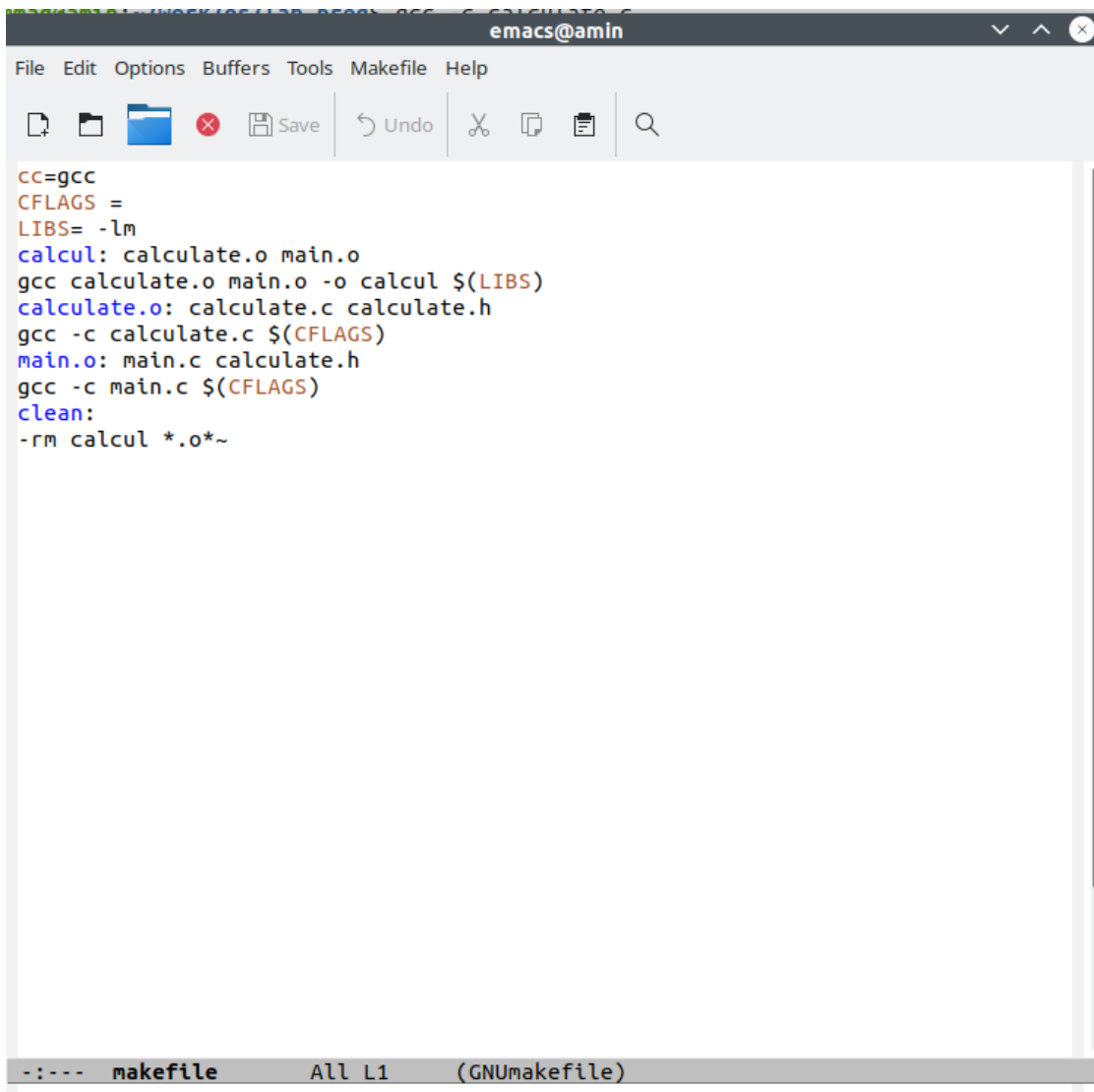
1. Выполнена компиляция программы посредством gcc. (см. рис. 5).



```
mokhammad@amin:~/work/os/lab_prog$ gcc -c calculate.c
mokhammad@amin:~/work/os/lab_prog$ gcc -c main.c
main.c: In function 'main':
main.c:13:9: warning: format '%s' expects argument of type 'char *', but argument 2 has type 'char (*)[4]' [-Wformat=]
  13 | scanf("%s",& Operation);
      |         ^~
      |         |
      |         | char (*)[4]
      |         char *
mokhammad@amin:~/work/os/lab_prog$ gcc -c calculate.o main.o -o calcul -lm
gcc: warning: calculate.o: linker input file unused because linking not done
gcc: warning: main.o: linker input file unused because linking not done
```

2. При необходимости исправлены синтаксические ошибки.

3. Создан Makefile. (см. рис. 6).



```
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*
```

-:--- makefile All L1 (GNUmakefile)

4. 6 С помощью gdb выполнена отладка программы calcul. (см. рис. 7, 8).

```
mokhammad@amin:~/work/os/lab_prog$ gdb ./calcul
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
Copyright (C) 2020 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:
<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 in ./calcul)
(gdb)
```

```

mokhammad@amin:~/work/os/lab_prog$ gdb ./calcul
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
Copyright (C) 2020 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:
<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 in ./calcul)
(gdb) run
Starting program: /home/mokhammad/work/os/lab_prog/calcul
Число: 3
Операция (+,-,*,/,pow,sqrt,sin,cos,tan): +
Второе слагаемое: 12
  15.00
[Inferior 1 (process 5730) exited normally]
(gdb) clear
No source file specified.
(gdb) list
1      init-first.c: No such file or directory.
(gdb) list 12,15
12      in init-first.c
(gdb) list calculate.c:20,29
No source file named calculate.c.
(gdb) list calculate.c:20,27
No source file named calculate.c.

```

7. 6. С помощью утилиты splint проанализированы коды файлов calculate.c и main.c. (см. рис. 9).

```

mokhammad@amin:~/work/os/lab_prog$ splint calculate.c
Splint 3.1.2 --- 20 Feb 2018

calculate.h:3:37: 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:6:31: Function parameter Operation declared as manifest array (size
constant is meaningless)
calculate.c: (in function Calculate)
calculate.c:12:7: 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:18:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:24:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:30:7: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:31:10: 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:34:10: Return value type double does not match declared type float:
(HUGE_VAL)
To allow all numeric types to match, use +relaxtypes.
calculate.c:42:4: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:43:10: Return value type double does not match declared type float:
(pow(Numeral, SecondNumeral))
calculate.c:46:8: Return value type double does not match declared type float:
(sqrt(Numeral))
calculate.c:48:8: Return value type double does not match declared type float:
(sin(Numeral))
calculate.c:50:8: Return value type double does not match declared type float:
(cos(Numeral))
calculate.c:52:13: Return value type double does not match declared type float:
(tan(Numeral))
calculate.c:56:11: Return value type double does not match declared type float:
(HUGE_VAL)

Finished checking --- 15 code warnings
mokhammad@amin:~/work/os/lab_prog$

```

## Вывод:

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

## контрольные вопросы

1. Как получить информацию о возможностях программ gcc, make, gdb и др.? Дополнительную информацию о этих программах можно получить с помощью функций info и man
2. Назовите и дайте краткую характеристику основным этапам разработки приложений в UNIX. Unix поддерживает следующие основные этапы разработки приложений: -создание исходного кода программы;- представляется в виде файла -сохранение различных вариантов исходного текста; -анализ исходного текста; Необходимо отслеживать изменения исходного кода, а также при работе более двух программистов над проектом программы нужно, чтобы они не делали изменений кода в одно время. -компиляция исходного текста и построение исполняемого модуля; -тестирование и отладка;

- Проверка кода на наличие ошибок -сохранение всех изменений, выполняемых при тестировании и отладке. 3. Что такое суффикс в контексте языка программирования? Приведите примеры использования. Использование суффикса ".c" для имени файла с программой на языке Си отражает удобное и полезное соглашение, принятое в ОС UNIX. Для любого имени входного файла суффикс определяет какая компиляция требуется. Суффиксы и префиксы указывают тип объекта. Одно из полезных свойств компилятора Си — его способность по суффиксам определять типы файлов. По суффиксу .c компилятор распознает, что файл abcd.c должен компилироваться, а по суффиксу .o, что файл abcd.o является объектным модулем и для получения исполняемой программы необходимо выполнить редактирование связей. Простейший пример командной строки для компиляции программы abcd.c и построения исполняемого модуля abcd имеет вид: gcc -o abcd abcd.c. Некоторые проекты предпочитают показывать префиксы в начале текста изменений для старых (old) и новых (new) файлов. Опция — prefix может быть использована для установки такого префикса. Плюс к этому команда bzr diff -p1 выводит префиксы в форме которая подходит для команды patch -p1. 8 Каково основное назначение компилятора языка C в UNIX? Основное назначение компилятора с языка Си заключается в компиляции всей программы в целом и получении исполняемого модуля. 9 Для чего предназначена утилита make? Описание взаимосвязей и соответствующих действий хранится в так называемом make-файле, который по умолчанию имеет имя makefile или Makefile. В общем случае make-файл содержит последовательность записей (строк), определяющих зависимости между файлами. Первая строка записи представляет собой список целевых (зависимых) файлов, разделенных пробелами, за которыми следует двоеточие и список файлов, от которых зависят целевые. 6. Приведите пример структуры Makefile. Дайте характеристику основным эле- ментам этого файла. Текст, следующий за точкой с запятой, и все последующие строки, начинающиеся с литеры табуляции, являются командами ОС UNIX, которые необходимо выполнить для обновления целевого файла. Таким образом, спецификация взаимосвязей имеет формат: target1 [ target2...]: [ ] [dependence1...] [(tab)commands] [#commentary] [(tab)commands] [#commentary], где # — специфицирует начало комментария, так как содержимое строки, начиная с # и до конца строки, не будет обрабатываться командой make; ; — последовательность команд ОС UNIX должна содержаться в одной строке make-файла (файла описаний), есть возможность переноса команд ( ), но она считается как одна строка; :: — последовательность команд ОС UNIX может содержаться в нескольких последовательных строках файла описаний. 8. Назовите и дайте основную характеристику основным командам отладчика gdb. — backtrace — выводит весь путь к текущей точке останова, то есть названия всех функций, начиная от main(); иными словами, выводит весь стек функций; — break — устанавливает точку останова; параметром может быть номер строки или название функции; — clear — удаляет все точки останова на текущем уровне стека (то есть в текущей функции); — continue — продолжает выполнение программы от текущей точки до конца; — delete — удаляет точку останова или контрольное выражение; — display — добавляет выражение в список выражений, значения кото- рых отображаются каждый раз



при остановке программы; – finish – выполняет программу до выхода из текущей функции; отображает возвращаемое значение, если такое имеется; – info breakpoints – выводит список всех имеющихся точек останова; – info watchpoints – выводит список всех имеющихся контрольных выражений; – list – выводит исходный код; в качестве параметра передаются название файла исходного кода, затем, через двоеточие, номер начальной и конечной строки; – next – пошаговое выполнение программы, но, в отличие от команды step, не выполняет пошагово вызываемые функции; – print – выводит значение какого-либо выражения (выражение передается в качестве параметра); – run – запускает программу на выполнение; – set – устанавливает новое значение переменной – step – пошаговое выполнение программы; – watch – устанавливает контрольное выражение, программа остановится, как только значение контрольного выражения изменится; 10. Прокомментируйте р