

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 1 and 2

Title: **Introduction and Variables**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Data Types

Variables and its types

Code and Output (Analysis)

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have install MinGW-w64, you need to configure it.

1. In the Windows search bar, type 'settings' to open your Windows Settings.
2. Search for Edit environment variables for your account.
3. Choose the Path variable and then select Edit.
4. Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
5. Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or powershell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

Basic Structure of C programming

A C Program starts with a main function and execute instruction present inside it.

Each instruction is terminated with semi colon (;).

Rules

- Program execution start from main ()
- All statements terminate with semicolon (:)
- Instructions are case-sensitive.
- Instructions are executed in the same order in which they are written.

Header Files

```
#include<stdio.h>
```

#include the pre-processor expands the line in turn and includes a copy of the standard header file `stdio.h` in the code as well. We have included `stdio.h` because we are using `print ()` (used to print text or value in c program).

Main Function

Syntax:

```
int main () {  
    *code*  
    return 0;  
}
```

From this function the execution of code starts. All the statement, declaration, function call, are written here.

Comments in C

Comments are those pieces of lines that are neglected by compiler during execution. Comments help us to understand other code.

- **Single Line Comment**

```
//This is single line comment. It starts with //.
```

- **Multi Line Comment**

```
/*  
  
This is  
  
Multi Line  
  
Comments  
  
*/
```

Data Types In C

In C, we have 3 data types:

1. int
2. float
3. char

Int refer to integer. It is used to store the integer number. For examples: 1, 2, 6, 40, etc

Float refer to floating point number. It is used to store the floating-point number (Number with decimal value). For examples: 1.23, 1.56, 6.25, etc.

Char refers to character. It is used to store character. For examples: a, u, w, %, &, etc.

Variables in C and its Types

Variable are the container which stores the value. Just like we have bottle to store water, container to store rice, daal, etc. In the same manner, we have different container in C to store different items.

A = 56

b = 4.7

c = 'A'

Before assigning value to a variable, we must declare its data types:

int A

float b

char c

Rules for Naming the Variables

- First character must be an alphabet or underscore (-)
- No commas, blanks are allowed in variable name
- No special symbol other than (-) is allowed
- Variable name is case sensitive.

i.e., Name and name are different

Getting start with C

```
#include <stdio.h>

int main(void)

{

printf("Hello C Program");

return 0;

}
```

Dissection of Program

```
#include<stdio.h>
```

Lines begin with # is used to communicate with the preprocessor. This #include line causes the preprocessor to include a copy of the header files studio.h at this point of code. The angular bracket<> indicated that it is to be found in usual place, which is system-dependent. We have included this file because it contains the information about the printf() function.

```
int main(void)
```

This is first line of function definition of main ().(We write parentheses after the name main to remind the reader that main () is a function.) The words int and void are keywords, also called reserved words which have special meaning in C. There are 32 reserved words in C.

```
int main(void)

{.....
```

Every program has function main (). Program execution always starts with this function. The top line should be read as "**main () is a function that takes no arguments and returns an int value.**" Here, the keyword int tells the compiler that this function returns a value of type int. The parentheses following main indicate to the compiler that main is a function. The keyword void indicates to the compiler that this function takes no arguments.

```
{
```

Braces surround the body of a definition. They are used to group statements together.

```
printf()
```

The C system contains a standard library of functions that can be used in programs. This is a function from the library that prints on the screen. We included the header file `stdio.h` because it provides certain information to the compiler about the function `printf()`.

```
"Hello C Program"
```

A string constant in C is a series of characters surrounded by double quotes. This string is an argument to the function `printf ()`, and it controls what gets printed.

```
printf("Hello C Program");
```

This is a call to the `printf()` function. In a program, the name of a function followed by parentheses causes the function to be called. It prints its argument, a string constant, on the screen.

```
return 0;
```

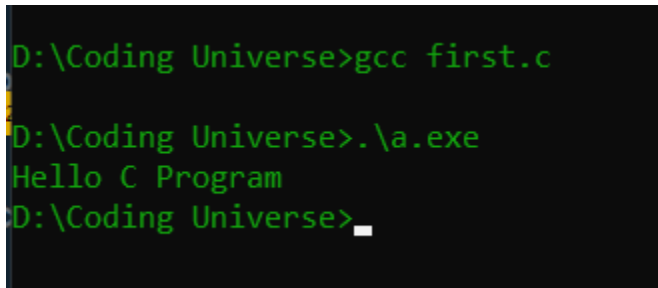
This is a return statement. It causes the value zero to be returned to the operating system. Our use of this return statement keeps the compiler happy. If we do not use it, the compiler will complain.

```
}
```

The right brace matches the left brace above, ending the function definition for `main ()`.

Running the program

- Save it as `first.c` and open command prompt on the folder where you have the file
- Run `gcc first.c`
- Then `.\a.exe`



```
D:\Coding Universe>gcc first.c
D:\Coding Universe>.\a.exe
Hello C Program
D:\Coding Universe>
```

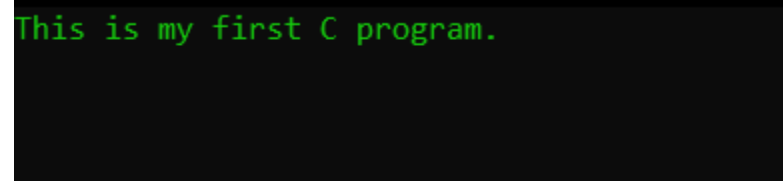

Lab 1

1.Type the following program and see the output

Source Code:

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
    printf("This is my first C program.");
    getch();
}
```

Output:



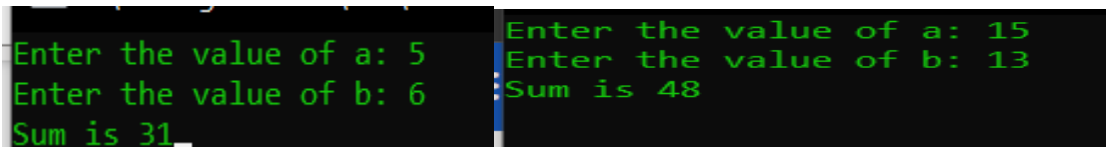
```
This is my first C program.
```

2.Type the following program and run with different input.

Source Code:

```
#include <stdio.h>
#include <conio.h>
int main (){
    int s, a, b, c = 20;
    printf("Enter the value of a: ");
    scanf("%d", &a);
    printf("Enter the value of b: ");
    scanf("%d", &b);
    s = a + b + c;
    printf("Sum is %d", s);
    getch();
    return 0;
}
```

Output:



```
Enter the value of a: 5
Enter the value of b: 6
Sum is 31

Enter the value of a: 15
Enter the value of b: 13
Sum is 48
```

3.Type the following program and run and see the output.

Source Code:

```
#include <stdio.h>
#include <conio.h>
void main ()
{
    int s, a, b;
    float p;
    system("cls");
    printf("Address of s is %x\n", &s);
    printf("Address of a is %x\n\n", &a);
    printf("Occupied number of bytes by variable s is %d\n\n\n", sizeof(s));
    printf("Size of a %d\n\n", sizeof(p));
    printf("Size of a %d\n\n\n\n\n\n", sizeof(1.5));
    printf("Size of floast data type is %d", sizeof(float));
    getch();
}
```

Output:

```
Address of s is 61fe1c
Address of a is 6422040

Occupied number of bytes by variable s is 4

Size of a 4
Size of a 8

Size of floast data type is 4
```

4. Write a program to calculate the area, circumference of a circle of radius r.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>

void main ()
{
    float area, circumference, r;

    system("cls")

    printf("Radius of circle:");
    scanf("%f", &r);

    area = M_PI * pow (r, 2);
    circumference = 2 * M_PI * r;

    printf("\nArea of Circle: %.2f m^2\n", area);
    printf("\nCircumference of circle: %.2f m.", circumference);

    getch();
}
```

Output:

```
Radius of circle:5  
Area of Circle: 78.54 m^2  
Circumference of circle: 31.42 m.
```

5. Write a program to calculate the volume of sphere of radius r.

Source code:

```
#include <stdio.h>  
#include <conio.h>  
#include <math.h>  
#include <stdlib.h>  
void main ()  
{  
    float volume, radius;  
    system("cls");  
    printf("Radius of Sphere: ");  
    scanf("%f", &radius);  
    volume = (float)4 / 3 * M_PI * pow (radius, 3);  
    printf("\n\nThe volume of the sphere is: %.2f m^3", volume);  
    getch();  
}
```

Output:

```
Radius of Sphere: 56
```

```
The volume of the sphere is:735618.63 m^3
```

6. Write a program to calculate the simple interest. Read values of P, T, R

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main ()
{
    float principal, time, rate, simple_interest;
    system("cls");
    printf("Principal: ");
    scanf("%f", &principal);
    printf("Time in years: ");
    scanf("%f", &time);
    printf("Rate per annum: ");
    scanf("%f", &rate);
    simple_interest = (principal * time * rate) / 100;
    printf("\n\nThe simple interest is: Rs%.2f", simple_interest);
    getch();
}
```

Output:

```
Principal: 1500
Time in years: 3
Rate per annum: 7

The simple interest is: Rs315.00
```

7. Write a program to read values of x and y from the user and evaluate the expression $v = x^3 + y^2 - 100/x$.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>

void main ()
{
    float x, y, expression_value;

    system("cls");

    printf("Enter the value of x and y respectively:");

    scanf("%f%f", &x, &y);

    expression_value = pow(x, 3) * pow(y, 2) - (float)100 / x;

    printf("Expression:\n v=(x^3+y^2) -(100/x)");

    printf("\n\nThe value of the expression is: %.2f", expression_value);

    getch();
}
```

Output:


```
Enter four integers one by one:1
5
8
9
The mean of the given integers is: 5.75
```

8. Write a program to read four integers from the user and display mean of the numbers

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main ()
{
    int a, b, c, d;
    float mean;
    system("cls");
    printf("\nEnter four integers one by one:");
    scanf("%d%d%d%d", &a, &b, &c, &d);
    mean = (float) (a + b + c + d) / 4;
    printf("The mean of the given integers is: %.2f", mean);
    getch();
}
```

Output:

```
Enter the length:15
Enter the breadth:45
Enter the height:49

The volume of the cuboid is: 33075.00
```

9. Write a program to read l, b and h of a cuboid and display its volume.

Source code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main ()
{
    float l, b, h, volume;
    system("cls");
    printf("Enter the length:");
    scanf("%f", &l);
    printf("Enter the breadth:");
    scanf("%f", &b);
    printf("Enter the height:");
    scanf("%f", &h);
    volume = (l * b * h);
    printf("\nThe volume of the cuboid is: %.2f", volume);
    getch();
}
```

Output:

```
Enter the price of two pen: 15
Enter the price of five copies: 50
The total price after discount is: Rs58.50
```

10. Write a program to read price of two pens and five copies of same type and calculate the price after discounting 10%

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

void main ()
{
    float pen_price, copies_price, total_price, discount;
    system("cls");
    printf("Enter the price of two pen: ");
    scanf("%f", &pen_price);
    printf("Enter the price of five copies: ");
    scanf("%f", &copies_price);
    discount = 0.1;
    total_price = (pen_price + copies_price) * (1 - discount);
    printf("The total price after discount is: Rs%.2f", total_price);
    getch();
}
```

Output:

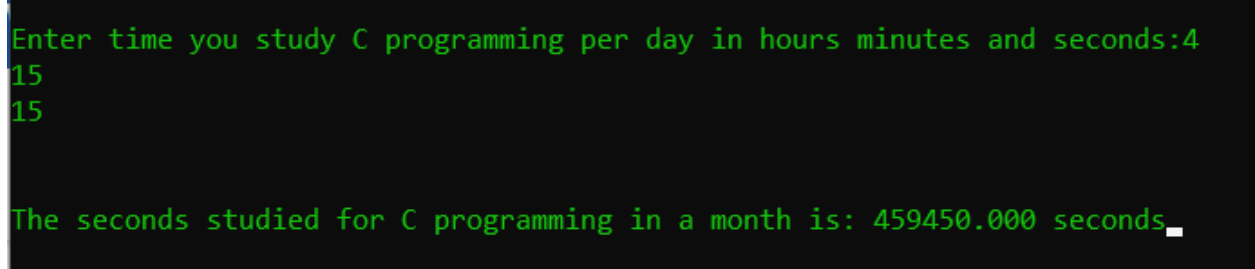
```
Enter the price of two pen: 56
Enter the price of five copies: 12
The total price after discount is: Rs61.20
```

11. Write a program to read time given for C programming study a day at your home in hours, minutes and seconds and display the total time in seconds in 30 days.

Source code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main()
{
    float h, m, s, month;
    system("cls");
    printf("\nEnter time you study C programming per day in hours minutes and seconds:");
    scanf("%f%f%f", &h, &m, &s);
    month = (h * 3600 + m * 60 + s) * 30;
    printf("\n\nThe seconds studied for C programming in a month is: %.3f seconds", month);
    getch();
}
```

Output:

A screenshot of a terminal window with a black background and green text. The text shows the program's execution: a prompt to enter time, followed by the user input '4 15 15', and the resulting output 'The seconds studied for C programming in a month is: 459450.000 seconds' followed by a cursor.

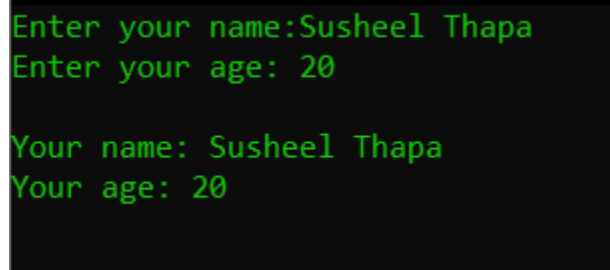
```
Enter time you study C programming per day in hours minutes and seconds:4
15
15

The seconds studied for C programming in a month is: 459450.000 seconds_
```

12. Write a program to read name, age of a person, and display them:

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main ()
{
    char name[40];
    int age;
    system("cls");
    printf("Enter your name:");
    gets(name);
    printf("Enter your age: ");
    scanf("%d", &age);
    printf("\nYour name: %s", name);
    printf("\nYour age: %d", age);
    getch();
}
```

Output:A screenshot of a terminal window with a black background and green text. The text shows the program's execution: 'Enter your name:Susheel Thapa', 'Enter your age: 20', followed by a blank line, and then 'Your name: Susheel Thapa' and 'Your age: 20' on the next line.

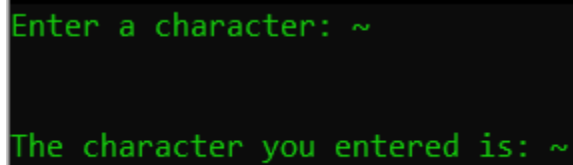
```
Enter your name:Susheel Thapa
Enter your age: 20

Your name: Susheel Thapa
Your age: 20
```

13. Write a program to read a character and display it.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main ()
{
    char x;
    system("cls");
    printf("Enter a character: ");
    scanf("%c", &x);
    printf("\n\nThe character you entered is: %c", x);
    getch();
}
```

Output:A screenshot of a terminal window with a black background and green text. The first line shows the prompt "Enter a character: ~" where '~' represents the user's input. The second line shows the output "The character you entered is: ~".

```
Enter a character: ~
The character you entered is: ~
```

Lab 2**1.Program that inputs seconds as input and convert to minutes.****Source Code:**

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
```

```

int                                seconds,                                min;

    system("cls");

    printf("Enter number of seconds: ");

scanf("%d", &seconds);

    min = seconds / 60; /*Integer Division*/

seconds = seconds % 60; /*Integer Division*/

    printf("\nMinutes= %d", min);

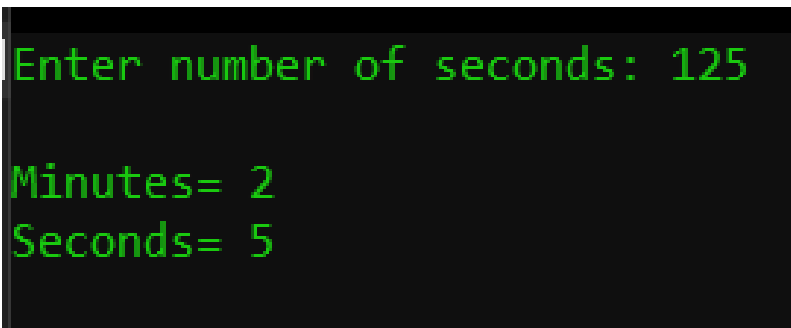
printf("\nSeconds= %d", seconds);

printf("Press any key to exit...");

getch();
}

```

Output:



```

Enter number of seconds: 125

Minutes= 2
Seconds= 5

```

2.A program to illustrate prefix increment operator

Source Code:

```

#include <stdio.h>

#include <conio.h>

void main ()

{

int x = 5, v;

```

```

system("cls");
v      =      ++x      *      ++x      +      ++x;
printf("v = %d, x = %d", v, x);
printf("Press any key to exit...");
getch();
}

```

Output



3.Program to read the three different integer from the user and display the largest among number them

Source code:

```

#include <stdio.h>
#include <conio.h>
void main () {
    int a, b, c, l;
    system("cls");
    printf("Enter the three different number: ");
    scanf("%d%d%d", &a, &b, &c);
    if (a > b && b > c) {
        l = a;
    }
    else if (b > c) {
        l = b;
    }
    else {

```

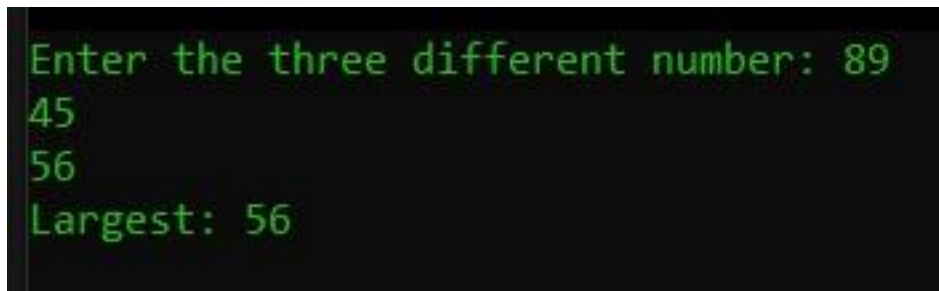


```

        l = c;
    }
    printf("Largest: %d", l);
    printf("Press any key to exit...");
    getch();
}

```

Output:



```

Enter the three different number: 89
45
56
Largest: 56

```

4.Type, Compile, run and observe and think about the output of the following program

Source code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

void main ()
{
    int x, y, z;

    system("cls");

    x = 30000, y = 20000;

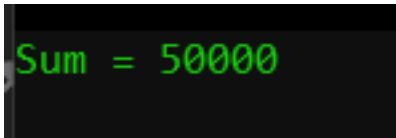
```

```

        z = x + y;
        printf("Sum = %d", z);
        printf("Press any key to exit...");
        getch();
    }

```

Output:



5. Type, run and observe the output of the following program

Source code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    float a;
    char b;
    long int c;
        unsigned int e;
        system("cls");
        printf("Enter the value of a: ");
        scanf("%f", &a);
        printf("Enter the value of b: ");
        scanf(" %c", &b);

```

```

printf("Enter the value of c and e: ");

scanf("%ld%u", &c, &e);

printf("Value of a: %f\nValue of b: %c\nValue of c: %ld\nValue of e: %u",
a, b, c, e);

getch();

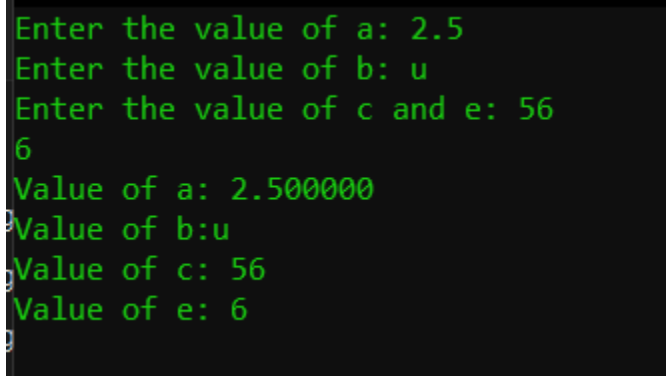
return 0;
}

```

No table of contents entries found.

No table of contents entries found.

Output:



```

Enter the value of a: 2.5
Enter the value of b: u
Enter the value of c and e: 56
6
Value of a: 2.500000
Value of b:u
Value of c: 56
Value of e: 6

```

6. Write a program to convert the given centigrade measure into Farenheit using relation

Source Code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main ()

```

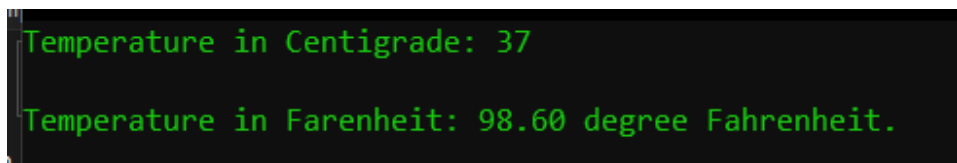
```

{
float fahrenheit, celsius;
system("cls");
printf("Temperature in Centigrade: ");
scanf("%f", &celsius);
fahrenheit = 1.8 * celsius + 32;
        printf("\nTemperature in Farenheit: %.2f degree Fahrenheit. ",
        fahrenheit);

getch();
return 0;
}

```

Output:



```

Temperature in Centigrade: 37
Temperature in Farenheit: 98.60 degree Fahrenheit.

```

7. Write a program to compute equivalent resistance of two resistor R1 and R2 when they are in series and in parallel

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
float resistance_one, resistance_two, resistance_series, resistance_parallel;
system("cls");

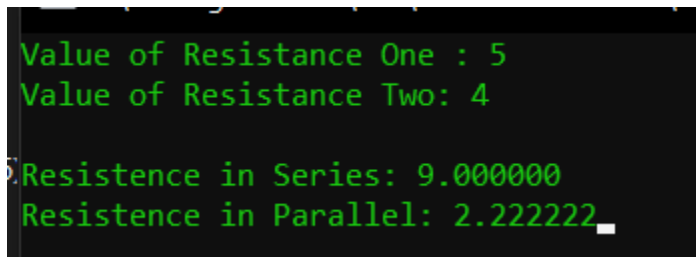
```

```

printf("Value of Resistance One: ");
scanf("%f", &resistance_one);
printf("Value of Resistance Two: ");
scanf("%f", &resistance_two);
resistance_series = resistance_one + resistance_two;
resistance_parallel = (resistance_one * resistance_two) / (resistance_one +
resistance_two);
printf("\nResistance in Series: %f\n", resistance_series);
printf("Resistance in Parallel: %f", resistance_parallel);
getch();
return 0;
}

```

Output:



```

Value of Resistance One : 5
Value of Resistance Two: 4
Resistance in Series: 9.000000
Resistance in Parallel: 2.222222_

```

8. Write a program to read two end points of a line, compute their mid point and display it.

Source code

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{

```

```

int x1, y1, x2, y2;
float mid_point_x, mid_point_y;
system("cls");
printf("X1: ");
scanf("%d", &x1);
printf("Y2: ");
scanf("%d", &y1);
printf("X2: ");
scanf("%d", &x2);
printf("Y2: ");
scanf("%d", &y2);
mid_point_x = (float)(x1 + x2) / 2;
mid_point_y = (float)(y1 + y2) / 2;
printf("The mid point of the line from (%d,%d) to (%d,%d) is (%.2f,%.2f).", x1, y1,
      x2, y2, mid_point_x, mid_point_y);
printf("Press any key to exit...");
getch();
return 0;
}

```

Output:

```

X1: 5
Y2: 9
X2: 4
Y2: 3
The mid point of the line from (5,9) to (4,3) is (4.50,6.00).

```

9. Write a program to read number of girls and boys in your class and display their ratio of girls to boys.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    int girls_number, boys_number;
    float ratio;
    system("cls");
    printf("Enter the number of girls: ");
    scanf("%d", &girls_number);
    printf("Enter the number of boys: ");
    scanf("%d", &boys_number);
    system("cls");
    ratio = ((float)girls_number) / ((float)boys_number);
    printf("No. of Girls: %d\n", girls_number);
    printf("No. of Boys: %d\n", boys_number);
    printf("\nRatio: %.2f #Girls to Boys\n", ratio);
    printf("Press any key to exit...");
    getch();
    return 0;
}
```

Output:

```
Enter the number of girls: 15
Enter the number of boys: 26

No. of Girls : 15
No. of Boys : 26

Ratio: 0.58   #Girls to Boys
```

10. Write a program to evaluate the following expression

$$S = x^5 + 0.2 * x * y + y^7$$

$$L = (a+b)^{(2x+y)/(p-q)} + c - 100$$

$$r = A/B \text{ [Where A and B are integers]}$$

$$R = ((u/x+v/y)^5)/((((p^2)/(3u^{2.5})) - (q/2v))^{3.5})$$

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
int main ()
{
    int x, y, a, b, p, q, c, A, B, u, v;
    float S, L, r, R;
    system("cls");
```



```

printf("Expression 1: \n\n\t S =x^5 + 0.2*x*y + y^7\n");
printf("\nValue of x: ");
scanf("%d", &x);
printf("Value of y: ");
scanf("%d", &y);
S = pow (x, 5) + (float)0.2 * x * y + pow (y, 7);
printf("\nValue of expression S =x^5 + 0.2*x*y + y^7 is %.3f ", S);

```

```

printf("\n\nExpression 2: \n\n\t L = (a+b)^((2x+y)/(p-q)) +c-100\n");
printf("\nValue of x: ");
scanf("%d", &x);
printf("Value of y: ");
scanf("%d", &y);
printf("Value of a: ");
scanf("%d", &a);
printf("Value of b: ");
scanf("%d", &b);
printf("Value of p: ");
scanf("%d", &p);
printf("Value of q: ");
scanf("%d", &q);

```

```

float base2 = (float) (a + b);
float power2 = (float)(2 * x + y) / (p - q);

```

```

L = pow (base2, power2) + c - 100;
printf("\nValue of expression L = (a+b)^((2x+y)/(p-q)) +c-100 is %.3f ", L);

printf("\n\nExpression 3: \n\n\t r = A/B [Where A and B are integers] \n");
printf("\nValue of A: ");
scanf("%d", &A);
printf("Value of B: ");
scanf("%d", &B);
r = A / B;
printf("\nValue of expression r = A/B [Where A and B are integers] without
      typecasting is %.2f ", r);
r = (float)A / B;
printf("\nValue of expression r = A/B [Where A and B are integers] with typecasting is
%.2f ", r);

printf("\n\nExpression 4: \n\n\t R = ((u/x+v/y) ^5)/((((p^2)/(3u^2.5)) -(q/2v)) ^3.5)
\n");
printf("\nValue of u: ");
scanf("%d", &u);
printf("Value of v: ");
scanf("%d", &v);
printf("Value of x: ");
scanf("%d", &x);
printf("Value of y: ");
scanf("%d", &y);
printf("Value of p: ");

```

```

scanf("%d", &p);
printf("Value of q: ");
scanf("%d", &q);
float numerator_base_4 = ((float)u / x + v / y);
float numerator = pow (numerator_base_4, 5);
float denominator_first_part = ((float)p * p) / (3 * pow (u, 2.5));
float denominator_second_part = (float)q / (2 * v);
float denominator_base = (denominator_first_part - denominator_second_part);
float denominator = pow(denominator_base, 3.5);
R = numerator / denominator;

printf("\nValue of expression R = ((u/x+v/y) ^5)/(((p^2)/(3u^2.5)) -(q/2v)) ^3.5 is
%.3f ", R);

getch();
return 0;
}

```

Output:

```

Expression 1:

    S =x^5 + 0.2*x*y + y^7

Value of x: 1
Value of y: 1

Value of expression S =x^5 + 0.2*x*y + y^7 is 2.200

Expression 2:

    L = (a+b)^((2x+y)/(p-q))+c-100

Value of x: 1
Value of y: 1
Value of a: 1
Value of b: 1
Value of p: 1
Value of q: 0

Value of expression L = (a+b)^((2x+y)/(p-q))+c-100 is -84.000

Expression 3:

    r = A/B [Where A and B are integers]

Value of A: 15
Value of B: 37

Value of expression r = A/B [Where A and B are integers] without typecasting is 0.00
Value of expression r = A/B [Where A and B are integers] with typecasting is 0.41

Expression 4:

    R = ((u/x+v/y)^5)/((((p^2)/(3u^2.5))-(q/2v))^3.5)

Value of u: 1
Value of v: 2
Value of x: 1
Value of y: 1
Value of p: 1
Value of q: 1

Value of expression R = ((u/x+v/y)^5)/((((p^2)/(3u^2.5))-(q/2v))^3.5) is 1454589.500000

```

11. Write a program to swap value of two variable

Source Code:

```
#include <stdio.h>
```

```

#include <conio.h>
#include <stdlib.h>
int main ()
{
int a, b, c;
system("cls");
printf("Value of a: ");
scanf("%d", &a);
printf("Value of b: ");
scanf("%d", &b);
system("cls");
printf("BEFORE SWAP\n\n");
printf("Value of a: %d \n", a);
printf("Value of b: %d \n", b);
c = a;
a = b;
b = c;

printf("\nAFTER SWAP\n\n");
printf("Value of a: %d \n", a);
printf("Value of b: %d \n", b);
printf("Press any key to exit...");
getch();
return 0;
}

```

Output:

```
Value of a: 15
Value of b: 16_
```

```
BEFORE SWAP
```

```
Value of a: 15
Value of b: 16
```

```
AFTER SWAP
```

```
Value of a: 16
Value of b: 15
_
```

Analysis and Discussion

In our first lab report here we have written a lot of programs and run it many times. Through this, we are able to know the respective syntax of the element like printf(), scanf(), gets () and many more.

Moreover, it provides us brief information about work flow of compiler, variable, data type, address, format specifier escape character, inbuilt keywords and many more.

Conclusion

Therefore the objective of the first lab session was completed with successful completion of the given exercise questions and better understanding of the aspects of the C language data types

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 1 and 2

Title: **Formatted I/O and Branching**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Formatted I/O

Branching

Code and Output

Algorithm

Flowchart

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

1. In the Windows search bar, type 'settings' to open your Windows Settings.
2. Search for Edit environment variables for your account.
3. Choose the Path variable and then select Edit.
4. Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
5. Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

Formatted Input

Integer

%wd --> w == Width of the int variable

Syntax:

```
scanf("%2d%3d", &a, &b);
```

Here the width of a is 2 and that of b is 3.

Float/Double

%wf --> w is width of float variable

Syntax:

```
scanf("%3f%4f", &a, &b);
```

This is also same as before the only difference is that previous one is integer and this one is float

Width of a is 3 and of b is 4.

Note that the '.' also occupy one width space

String

`%ws` --> `w` is the width of string

Syntax:

```
scanf("%12s", str);
```

This set the field of `str` to 12. And `str` cannot have more than 12 width space or

Character

Read string in C using `scanf()` with `%c`

```
scanf("%10c", str);
```

It is not a good idea to use this method. The above program read exactly 9 characters if we try to give less than 9 characters than it will not give output until at least 9-character size is reached.

Read string in C using scan set conversion code (`[...]`)

```
scanf("%[a-z]",str);
```

This reads lowercase characters whenever it encounters another character except for lowercase charcater then it stops reading

Read string in C using `scanf` with `[\n]` (single line)

```
scanf("%[\n]", str);
```

The circumflex (^) plays an important role while taking input. `[\n]` will read input until the user presses the enter key, otherwise, it will keep reading input from the screen.

Multiline input using `scanf`

```
scanf("%[^\n]", str);
```

The above program will read input until the user enters the star * symbol. After * symbol, it will stop reading. In this way, we can read multiple lines of characters. It will read white space also, only stop reading when a * symbol came. Otherwise, it will keep reading the input from the user.

Formatted Output

Integer Output

`%wd:`

Here ‘d’ is used in its standard meaning while ‘w’ specifies the integer number which represents the minimum field width of any integer output data.

Floating Point Output

`%w.nf:`

Here ‘f’ is used in its standard meaning to represent the floating-point number, while ‘w’ specifies the integer number which represents the minimum field width of any integer output data. Here ‘n’ specifies the number of digits after the decimal. By default, the six digits have to be printed after the decimal.

String Output

`%w.ns:`

In formatted string output the decimal point and the latter ‘n’ are used optionally. Here ‘n’ specifies the first ‘n’ characters of the string that will be displayed along with the ‘w-n’ leading blanks.

Branching

If...else statement

Syntax:

```
if(boolean_expression)  
{ /* statement(s) will execute if the Boolean expression is true */}  
else  
{ /* statement(s) will execute if the Boolean expression is false */}
```

If the Boolean expression evaluates to true, then the if block will be executed, otherwise, the else block will be executed.

If...else if...else Statement

An if statement can be followed by an optional else if...else statement, which is very useful to test various conditions using single if...else if statement.

When using if...else if..else statements, there are few points to keep in mind –

- An if can have zero or one else's and it must come after any else ifs.
- An if can have zero to many else if's and they must come before the else.
- Once an else if succeeds, none of the remaining else if's or else's will be tested.

Syntax:

```
if(boolean_expression 1)  
{ /* Executes when the boolean expression 1 is true */}  
else if( boolean_expression 2)  
{ /* Executes when the boolean expression 2 is true */}  
else if( boolean_expression 3)  
{ /* Executes when the boolean expression 3 is true */}  
else  
{ /* executes when the none of the above condition is true */}
```

For Loop

A for loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

Syntax:

```
for ( init; condition; increment)  
{ statement(s);}
```

Here is the flow of control in a 'for' loop –

- The init step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
- Next, the condition is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and the flow of control jumps to the next statement just after the 'for' loop.
- After the body of the 'for' loop executes, the flow of control jumps back up to the increment statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.
- The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again condition). After the condition becomes false, the 'for' loop terminates.

While Loop

A while loop in C programming repeatedly executes a target statement as long as a given condition is true.

Syntax

```
while(condition)  
{ statement(s);}
```

Here, statement(s) may be a single statement or a block of statements. The condition may be any expression, and true is any nonzero value. The loop iterates while the condition is true.

When the condition becomes false, the program control passes to the line immediately following the loop.

Do ...While Loop

A do...while loop is similar to a while loop, except the fact that it is guaranteed to execute at least one time.

Syntax:

```
do { statement(s);  
while( condition );
```

The conditional expression appears at the end of the loop, so the statement(s) in the loop executes once before the condition is tested.

If the condition is true, the flow of control jumps back up to do, and the statement(s) in the loop executes again. This process repeats until the given condition becomes false.

CODE AND OUTPUT

Lab 3

1. Write a program to read a character using getch()/getchar() and display using putchar()/putchar()

Algorithm:

- Start
- Declare a variable
- Read the value of variable
- Display the variable
- End

Source code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    char a_getch, a_getchar;
    system("cls");
    printf("Enter another character: ");
    a_getchar = getchar();
    printf("Enter any character: ");
    a_getch = getch(); /*This getch wonot be display in termianl as you type*/
    printf("\nValue taken via getch is ");
    putchar(a_getchar);
    printf("\nValue taken via getch is ");
    putchar(a_getch);
    getch();
    return 0;
}
```

Output:

```
Enter another character: F
Enter any character:
Value taken via getchar is F
Value taken via getch is * _
```

2. Write a program to read a character and string using scanf() and display it using printf().

Algorithm:

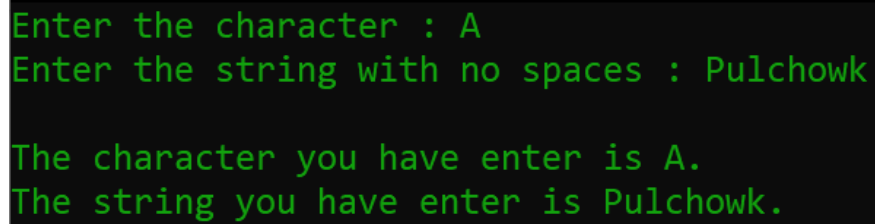
- Start
- Declare a variable
- Read the value of variable (scanf())
- Print the readed variable value
- End

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    int character, string[20];
    system("cls");
    printf("Enter the character : ");
    scanf("%c", &character);
    printf("Enter the string with no spaces : ");
    scanf("%s", string);
    printf("\nThe character you have enter is %c.", character);
```

```
printf("\nThe string you have enter is %s.", string);
getch();
return 0;
}
```

Output:

A screenshot of a terminal window with a black background and green text. It shows the output of a C program. The first two lines are prompts: "Enter the character : A" and "Enter the string with no spaces : Pulchowk". The next two lines are the program's output: "The character you have enter is A." and "The string you have enter is Pulchowk.".

```
Enter the character : A
Enter the string with no spaces : Pulchowk

The character you have enter is A.
The string you have enter is Pulchowk.
```

3. Write a program to read a string using gets() and display using puts()

Algorithm:

- Start
- Declare a string variable
- Read the value of string and store in the declared variable (gets ())
- Display the variable value
- End

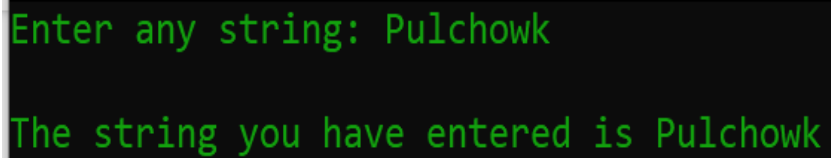
Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    char string_gets[90];
    system("cls");
    printf("Enter any string: ");
    gets(string_gets);
    printf("\nThe string you have entered is ");
    puts(string_gets);
}
```

```
getch();  
return 0;  
}
```

Output:



```
Enter any string: Pulchowk  
The string you have entered is Pulchowk
```

4.This illustrates different format specifications for printing integer numbers.

Source Code:

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
int main()  
{  
int a = 12345;  
system("cls");  
printf("\nCase 1: %d", a);  
printf("\nCase 2: %i", a);  
printf("\nCase 3: %15d", a);  
printf("\nCase 4: %-15d", a);  
printf("\nCase 5: %015d", a);  
printf("\nCase 6: %+15d", a);  
printf("\nCase 7: %3d", a);  
getch();  
return 0;  
}
```

Output:

```
Case 1: 12345
Case 2: 12345
Case 3:          12345
Case 4: 12345
Case 5: 00000000012345
Case 6: +12345
Case 7: 12345
```

5.This example illustrates different format specifications for printing real numbers.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
float n = 123.9876;
system("cls");
printf("\nCase 1: %f", n);
printf("\nCase 2: %e", n);
printf("\nCase 3: %g", n);
printf("\nCase 4: %15.4d", n);
printf("\nCase 5: %-15.3d", n);
printf("\nCase 6: %015.4ed", n);
printf("\nCase 7: %.8d", n);
printf("\nCase 8: %2.2d", n);
getch();
return 0;
```

```
}
```

Output:

```
Case 1: 123.987602
Case 2: 1.239876e+002
Case 3: 123.988
Case 4:      -536870912
Case 5: -536870912
Case 6: 00001.2399e+002d
Case 7: -536870912
Case 8: -536870912
```

6.This example illustrate different format specifier for printing character

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    char ch = 'a';
    system("cls");
    printf("\nCase 1: %c", ch);
    printf("\nCase 2: %10c", ch);
    printf("\nCase 3: %-10c", ch);
    getch();
    return 0;
}
```

Output:

```
Case 1: a
Case 2:      a
Case 3: a
```

7.This example illustrate different format specifier for printing string

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    char str[20] = "I love Balgung.";
    system("cls");
    printf("\nCase 1: %s", str);
    printf("\nCase 2: %18s", str);
    printf("\nCase 3: %-18s", str);
    printf("\nCase 4: %18.8s", str);
    printf("\nCase 5: %-18.9s", str);
    printf("\nCase 6: %5s", str);
    printf("\nCase 7: %10s", str);
    getch();
    return 0;
}
```

Output:

```
Case 1: I love Balgung.
Case 2:      I love Balgung.
Case 3: I love Balgung.
Case 4:              I love B
Case 5: I love Ba
Case 6: I love Balgung.
Case 7: I love Balgung.
```

8.This example illustrates the concept of printing mixed data.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    int n = 12345;
    float m = 123.9876;
    char ch = 'a';
    char str[20] = "I love Baglung.";
    system("cls");
    printf("n=%7d\nm=%12.5f\nch=%-2c\nstr=%16s\n", n, m, ch, str);
    getch();
    return 0;
}
```

Output:


```
n= 12345
m= 123.98760
ch=a
str= I love Baglung.
```

9.This example illustrates different format specifications for reading integers numbers

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    int a, b;
    system("cls");
    printf("Enter the number: ");
    scanf("%d", &a);
    printf("The read and stored value of a is %d\n\n", a);
    printf("Enter another integers number: ");
    scanf("%3d", &b);
    printf("The read and stored value of b is %d\n", b);
    getch();
    return 0;
}
```

Output:

```
Enter the number: 123456789
The read and stored value of a is 123456789

Enter another integers number: 1234567
The read and stored value of b is 123
```

10.This example illustrates the concept of reading string using %wc format specification

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    char str[50];
    system("cls");
    printf("Enter the string: ");
    scanf("%10c", &str);
    printf("Read string : %s", str);
    getch();
    return 0;
}
```

Output:

```
Enter the string: iAmVeryGoodBoyLivesInChitwan
Read string : iAmVeryGoo
```

11.This example shows the concept of defining search set to read string.

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    char str[70];
    system("cls");
    printf("Enter the string: ");
    scanf("%[a-zA-Z0-9]",str);
    printf("Read string: %s",str);
    getch();
    return 0;
}

```

Output:

```

Enter the string: noiwjeq$
Read string: noiwjeq

```

```

Enter the string: ak.heo9nwekUW3*KAF
Read string: ak

```

12.This example show the concept of defining search set to read string

Source Code:

```

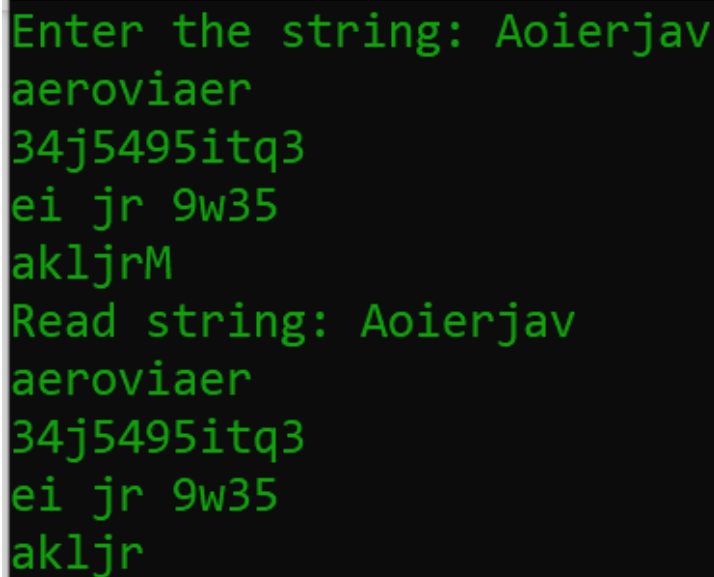
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{

```

```
char str[70];
system("cls");
printf("Enter the string: ");
scanf("%[^M]",str);
printf("Read string: %s",str);
getch();
return 0;
}
```

Output:

A screenshot of a terminal window with a black background and green text. The text shows the output of a C program. It starts with the prompt "Enter the string: " followed by the input "Aoierjav". Then, the prompt "Read string: " is shown, followed by the same input "Aoierjav". The input is repeated on the next line, and then the program ends with a newline character.

```
Enter the string: Aoierjav
aeroviaer
34j5495itq3
ei jr 9w35
akljrM
Read string: Aoierjav
aeroviaer
34j5495itq3
ei jr 9w35
akljr
```

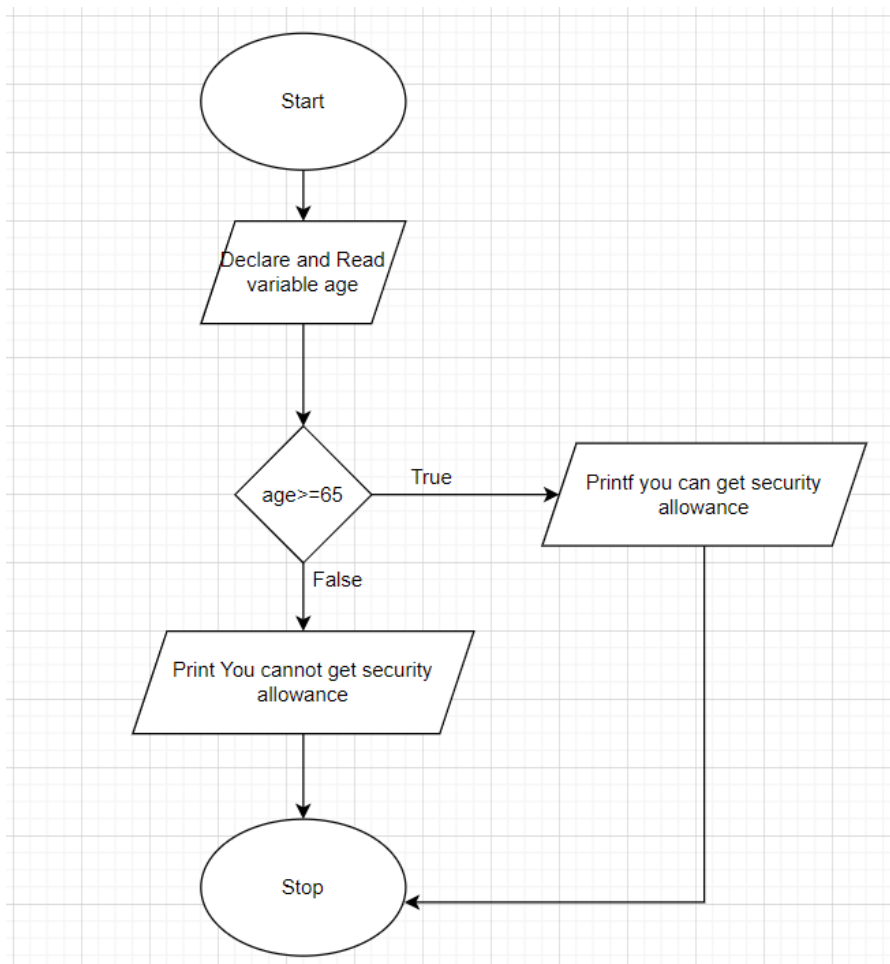
LAB 4

1.If a person age is greater than 65, he gets security allowance. Write a program to read the age of a person and display the appropriate message.

Algorithm:

- ❖ Start
- ❖ Declare a variable age
- ❖ Read the value of age and store the value in age
- ❖ if age>65
 - True: Print you will get security allowance
 - False: Print you won't get security allowance.
- ❖ Stop

Flowchart:



Source Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
int main(){
    int age;
    system("cls");
    printf("Enter your age: ");
    scanf("%d", &age);
    printf("\nYour age is %d.",age);
```

```
if(age >=65){  
    printf("\nYou can get security allowance.");  
}  
else{  
    printf("\nYou can't get security allowance.");  
}  
getch();  
return 0;  
}
```

Output:

```
Enter your age: 65  
  
Your age is 65.  
You can get security allowance.
```

```
Enter your age: 15  
  
Your age is 15.  
You can't get security allowance.
```

2. Write a program to read an integer from the user and check whether it is positive, zero or negative and display the appropriate message.

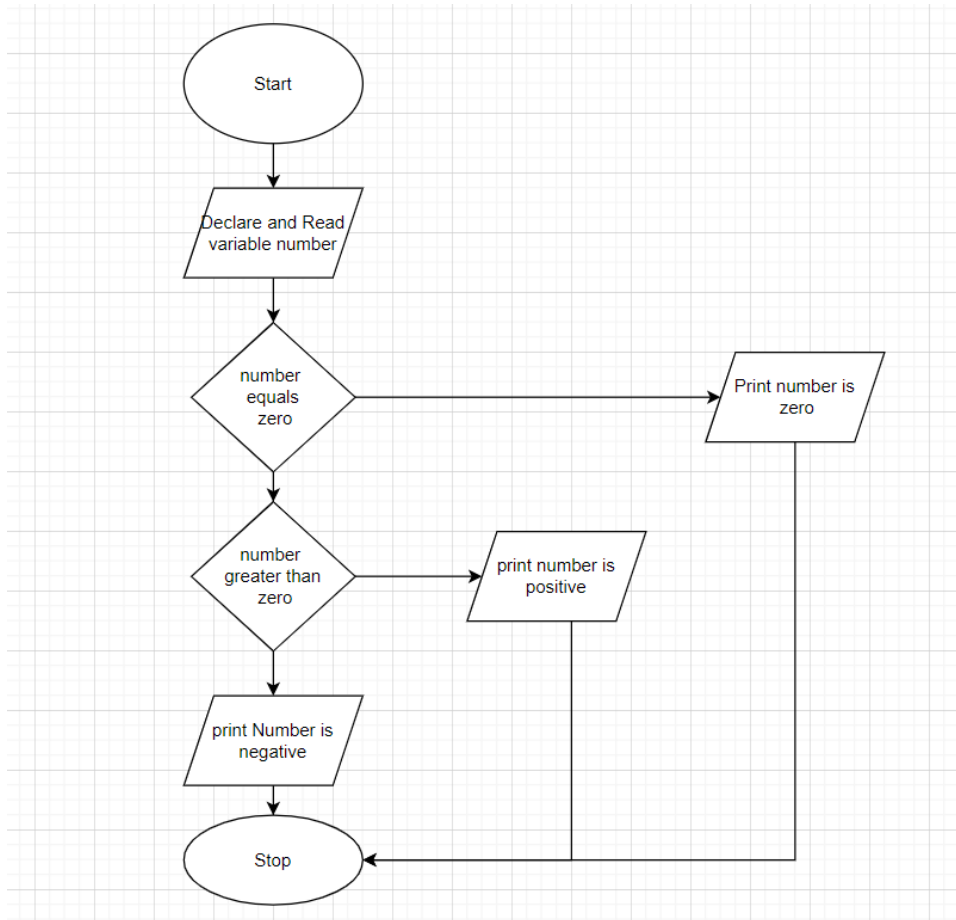
Algorithm:

- ❖ Start
- ❖ Declare a variable number
- ❖ Read the value of number
- ❖ Check number == 0
 - True: Print Number is zero
 - False:
 - Check number > 0
 - True: Print number is greater than zero

- False: Print number is less than zero.

❖ Stop

Flow Chart:



Source Code:

```
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
int main(){
    int number;
    system("cls");
    printf("Enter any number: ");
```

```
scanf("%d", &number);
printf("\nNumber = %d", number);
if (number == 0){
printf("\nIt is zero.");
}
else if(number>0){
printf("\nNumber is positive.");
}
else{
printf("\nNumber is negative.");
}
getch();
return 0;
}
```

Output:

```
Enter any number: 9
```

```
Number = 9
```

```
Number is positive.
```

```
Enter any number: -8
```

```
Number = -8
```

```
Number is negative. _
```

```
Enter any number: 0
```

```
Number = 0
```

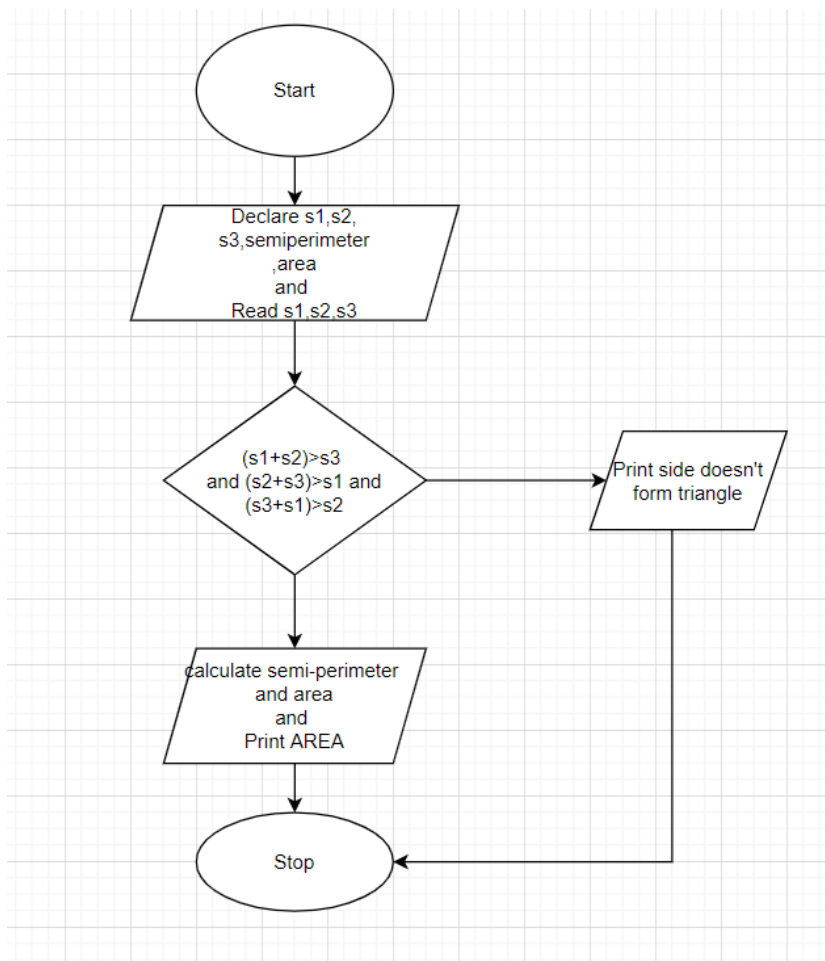
```
It is zero.
```


3. Write a program to read the three sides of triangle from the user and calculate the area of the triangle. Be sure to check the condition if triangle is formed or not.

Algorithm:

- ❖ Start
- ❖ Declare variable side one, side two, side three, area, semiperimeter
- ❖ Read the value of sides from user
- ❖ Check sum of any two sides > third side
 - True:
 - calculate semi-perimeter and then area
 - print the value of area
 - False:
 - Print error value don't form triangle
- ❖ Stop

Flow chart:



Source Code:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>

int main (){
float side_one, side_two, side_three, semi_perimeter, area_square, area;
system("cls");
printf("Side one: ");
scanf("%f",&side_one);
printf("Side two: ");
scanf("%f",&side_two);
printf("Side three: ");
scanf("%f",&side_three);
semi_perimeter = (side_one+ side_three+side_two)/2.0;
area_square = semi_perimeter*(semi_perimeter-side_one)*(semi_perimeter-
side_two)*(semi_perimeter-side_three);
area = pow(area_square, 0.5);
if((side_one+side_three)>side_two && (side_two+side_three)>side_one &&
(side_one+side_two)>side_three)
{
printf("\nThe area of the triangle is %.2f.", area);
}
else
{
printf("\nError: Side given were not of triangle. Check your value once.");
}
getch();
```

```
return 0;
}
```

Output:

```
Side one: 3
Side two: 4
Side three: 5

The area of the triangle is 6.00.
```

```
Side one: 1
Side two: 9
Side three: 12

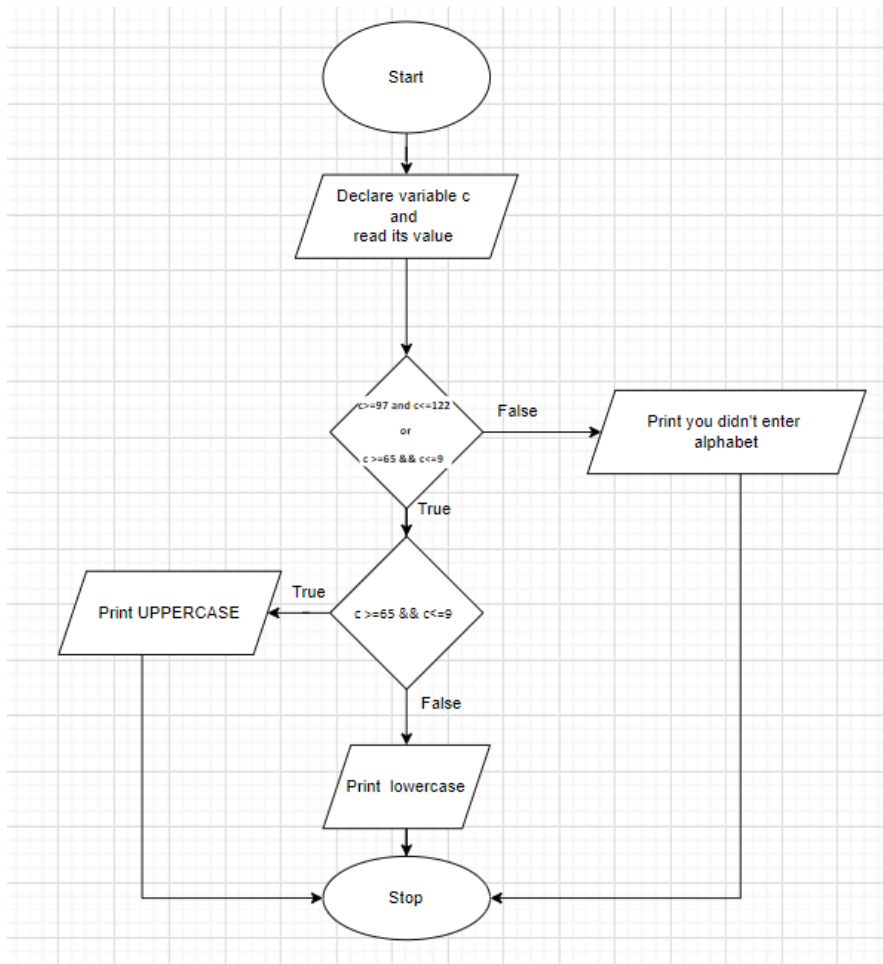
Error: Side given were not of triangle. Check your value once.
```

4. Write a program to read a character and check whether the character is upper or lower case

Algorithm:

- ❖ Start
- ❖ Declare the variable character
- ❖ Read the value of character
- ❖ $\text{character} \geq 97$ and $\text{character} \leq 122$ or $\text{character} \geq 65$ && $\text{character} \leq 90$
 - True:
 - $\text{character} \geq 65$ && $\text{character} \leq 90$
 - True:
 - Print it is upper case
 - False:
 - Print it is lower case
 - False:
 - Error!! You did not enter correct input
 - ❖ Stop

Flow chart:



Source Code:

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
int main(){
    char character;
    system("cls");
    printf("Enter an alphabet: ");
    scanf("%c", &character);
    printf("Alphabet: %c", character);
    if((character >= 97 && character <= 122) || (character >= 65 && character <= 90)){
        if (character >= 65 && character <= 90){

```

```

printf("\nIt is UPPER CASE");
}
else{
printf("\nIt is lower case.");
}
}
else{
printf("\nERROR: You didn't input alphabet.");
}
getch();
return 0;
}
/*

```

The ASCII value of the lowercase alphabet is from 97 to 122. And, the ASCII value of the uppercase alphabet is from 65 to 90. If the ASCII value of the character entered by the user lies in the range of 97 to 122 or from 65 to 90, that number is an alphabet.

**/*

Output:

```

Enter an alphabet: R
Alphabet: R
It is UPPER CASE_

```

```

Enter an alphabet: R
Alphabet: R
It is UPPER CASE_

```

```

Enter an alphabet: a
Alphabet: a
It is lower case._

```

```

Enter an alphabet: &
Alphabet: &
ERROR: You didn't input alphabet.

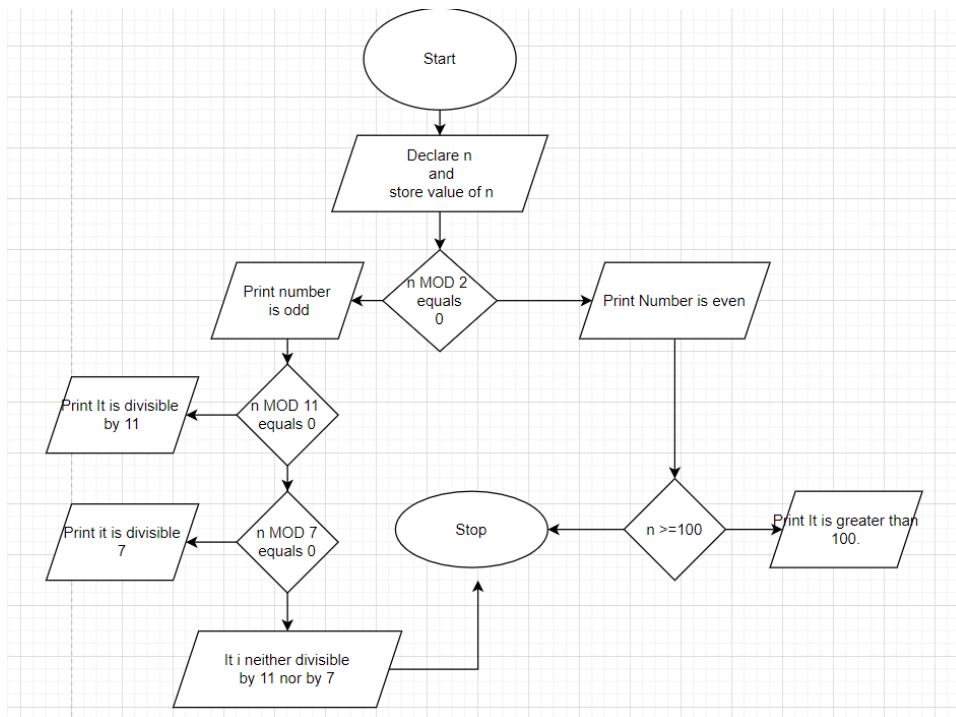
```

5. Write a program to read a unsigned integer and check whether the number is odd or even. If it is even , check whether it is greater than 100 or not and Display the message. If the number is odd, check whether it is divisible by 11 but by 7 and display the appropriate message

Algorithm:

- Start
- Declare variable number as store value given by user.
- Remainder when number / 2 equals to 0.
 - True :
 - print It is even number
 - check number > 100
 - True ;
 - print number is greater than 100.
 - False
 - print It is odd number.
 - remainder when number divided by 11 equals to 0
 - True :
 - Print number is divisible by 11
 - False :
 - remainder when number divided by 7 equals to 0
 - True :
 - number is divided by 7
 - False :
 - print Number is neither divisible by 11 not by 7
- Stop

Flow Chart:



Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    unsigned int number;
    system("cls");
    printf("Enter any number: ");
    scanf("%u", &number);
    printf("\nNumber : %u", number);
    if (number % 2 == 0)
    {
        printf("\nNumber is even number.");
    }
    if (number > 100)
    {

```

```
printf("\nNumber is greater than 100.");  
}  
}  
else  
{  
printf("\nNumber is odd number.");  
if (number % 11 == 0)  
{  
printf("\nNumber is divisible by 11.");  
}  
else if (number % 7 == 0)  
{  
printf("\nNumber is divisible by 7.");  
}  
else  
{  
printf("\nNumber isn't divisible by 11 or 7.");  
}  
}  
getch();  
return 0;  
}
```

Output:


```
Enter any number: 7

Number : 7
Number is odd number.
Number is divisible by 7.
```

```
Enter any number: 11

Number : 11
Number is odd number.
Number is divisible by 11.
```

```
Enter any number: 13

Number : 13
Number is odd number.
Number isn't divisible by 11 or 7.
```

```
Enter any number: 122

Number : 122
Number is even number.
Number is greater than 100.
```

6. Write a program to determine the root of quadratic equation. Take the value of a, b and c from user.

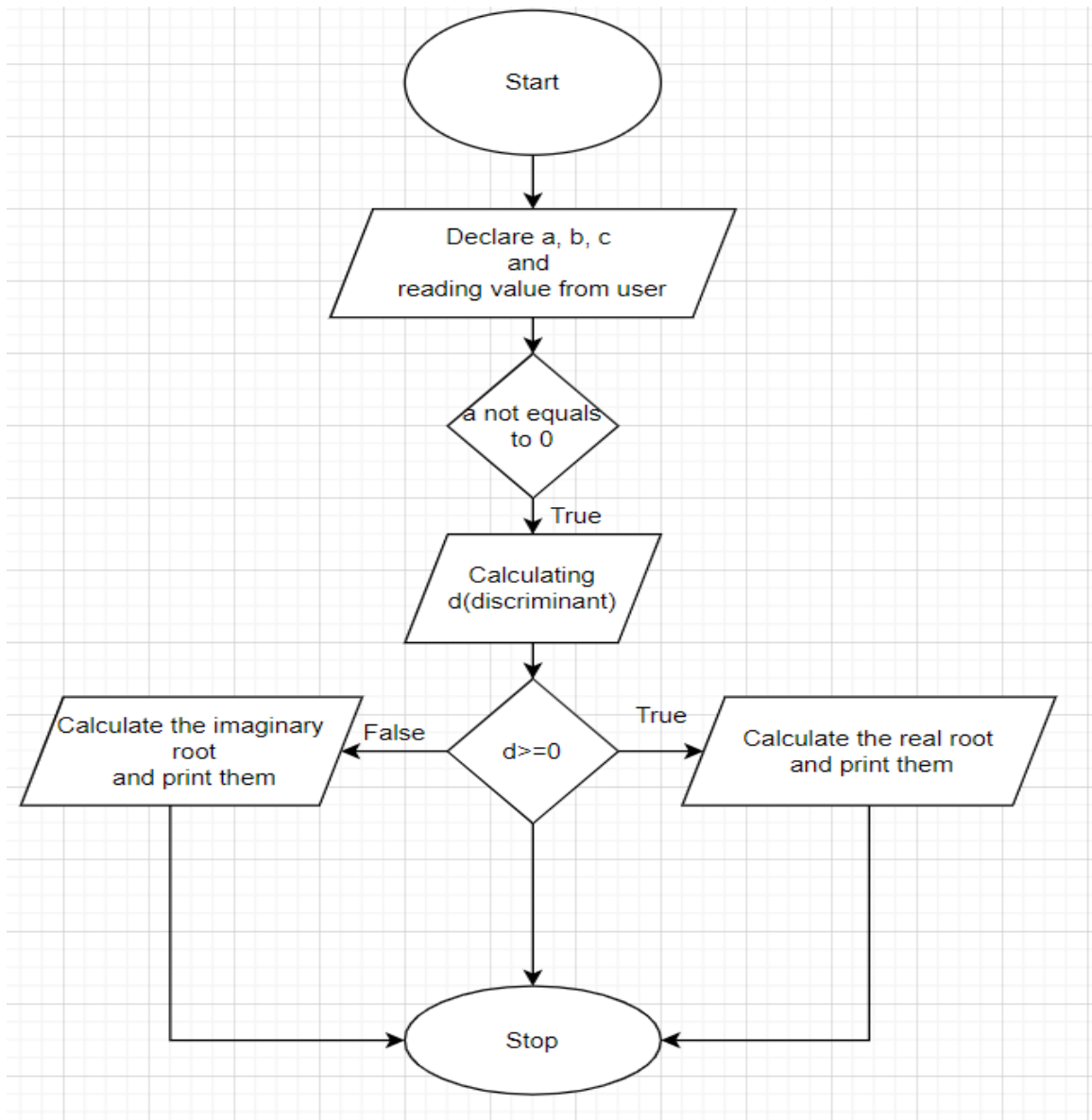
Algorithm:

- ❖ Start
- ❖ Declare a,b,c, and reading value of a,b,c
- ❖ Check a not equal to zero
 - True:
 - finding $d = \text{square root of } (b^2 - 4ab)$
 - Checking $d \geq 0$

- True:
 - ♦ calculating real root
- False:
 - ♦ Calculating imaginary root

❖ Stop

Flow Chart:



Source Code:

```
#include <stdio.h>
```

```

#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
float a, b, c, discriminant, root_one_real, root_two_real, root_imaginary;
system("cls");
printf("Enter the value of a,b,c:\n");
scanf("%f %f %f", &a, &b, &c);
if (a != (float)0)
{
discriminant = pow(b, 2) - 4 * a * c;
if (discriminant >= 0)
{
root_one_real = (-b + pow(discriminant, 0.5)) / (2 * a);
root_two_real = (-b - pow(discriminant, 0.5)) / (2 * a);
printf("\nRoot One : %.2f\nRoot Two : %.2f", root_one_real, root_two_real);
}
else if (discriminant < 0)
{
discriminant = -discriminant;
root_one_real = (-b) / (2 * a);
root_two_real = (-b) / (2 * a);
root_imaginary = (pow(discriminant, 0.5)) / (2 * a);
printf("\nRoot One : %.2f + %.2fi \nRoot Two : %.2f - %.2fi", root_one_real,
root_imaginary, root_two_real, root_imaginary);
}
}
}

```

```

else
{
printf("\nYou didnot give correct quadratic equation.");
}
return 0;
}

```

Output:

```

Enter the value of a,b,c:
1
2
1

Root One : -1.00
Root Two : -1.00

Enter the value of a,b,c:
1
2
9

Root One : -1.00 + 2.83i
Root Two : -1.00 - 2.83i

```

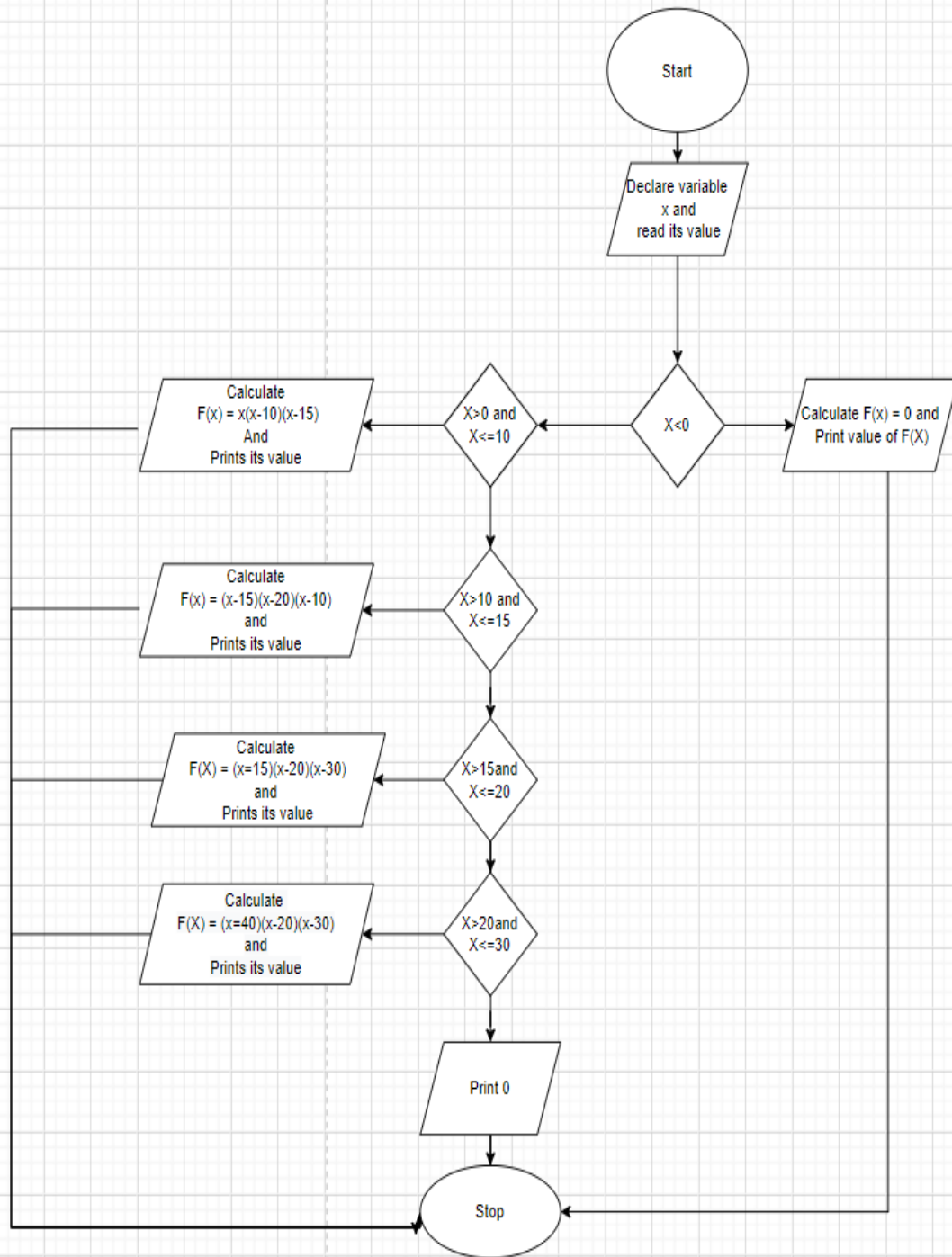
7. Write a program to evaluate the following function(F(x)) given by

Algorithm:

- Start
- Declare variable x and read its value
- $X < 0$
 - True:

- Calculate $f(x) = 0$ and prints $f(x)$
- False:
 - $X > 0$ and $X \leq 10$
 - True:
 - Calculate $f(x) = x(x-10)(x-15)$ and prints $f(x)$
 - False:
 - $X > 10$ and $X \leq 15$
 - True:
 - Calculate $f(x) = (x-20)(x-10)(x-15)$ and prints $f(x)$
 - False:
 - $X > 20$ and $X \leq 30$
 - True:
 - Calculate $f(x) = (x-20)(x-30)(x-40)$ and prints $f(x)$
 - False:
 - Print 0
 - Stop

Flow Chart:



Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()

```

```

{
float number, function_value;
system("cls");
printf("To calculate:\n \t\tF(x)\n\n");
printf("X = ");
scanf("%f", &number);
if (number <= 0)
{
function_value = 0;
}
else if (number > 0 && number <= 10)
{
function_value = number * (number - 10) * (number - 15);
}
else if (number > 10 && number <= 15)
{
function_value = (number - 10) * (number - 15) * (number - 20);
}
else if (number > 15 && number <= 20)
{
function_value = (number - 30) * (number - 15) * (number - 20);
}
else if (number > 20 && number <= 30)
{
function_value = (number - 30) * (number - 20) * (number - 40);
}
else
{

```

```
function_value = 0;
}
printf("\nThe value of function is %f.", function_value);
getch();
return 0;
}
```

Output:

```
To calculate:
           F(x)

X = -6

The value of function is 0.000000.
```

```
To calculate:
           F(x)

X = 2

The value of function is 208.000000.
```

```
To calculate:
           F(x)

X = 12

The value of function is 48.000000.
```



```
To calculate:
```

```
F(x)
```

```
X = 18
```

```
The value of function is 72.000000.
```

```
To calculate:
```

```
F(x)
```

```
X = 25
```

```
The value of function is 375.000000.
```

```
To calculate:
```

```
F(x)
```

```
X = 46
```

```
The value of function is 0.000000.
```

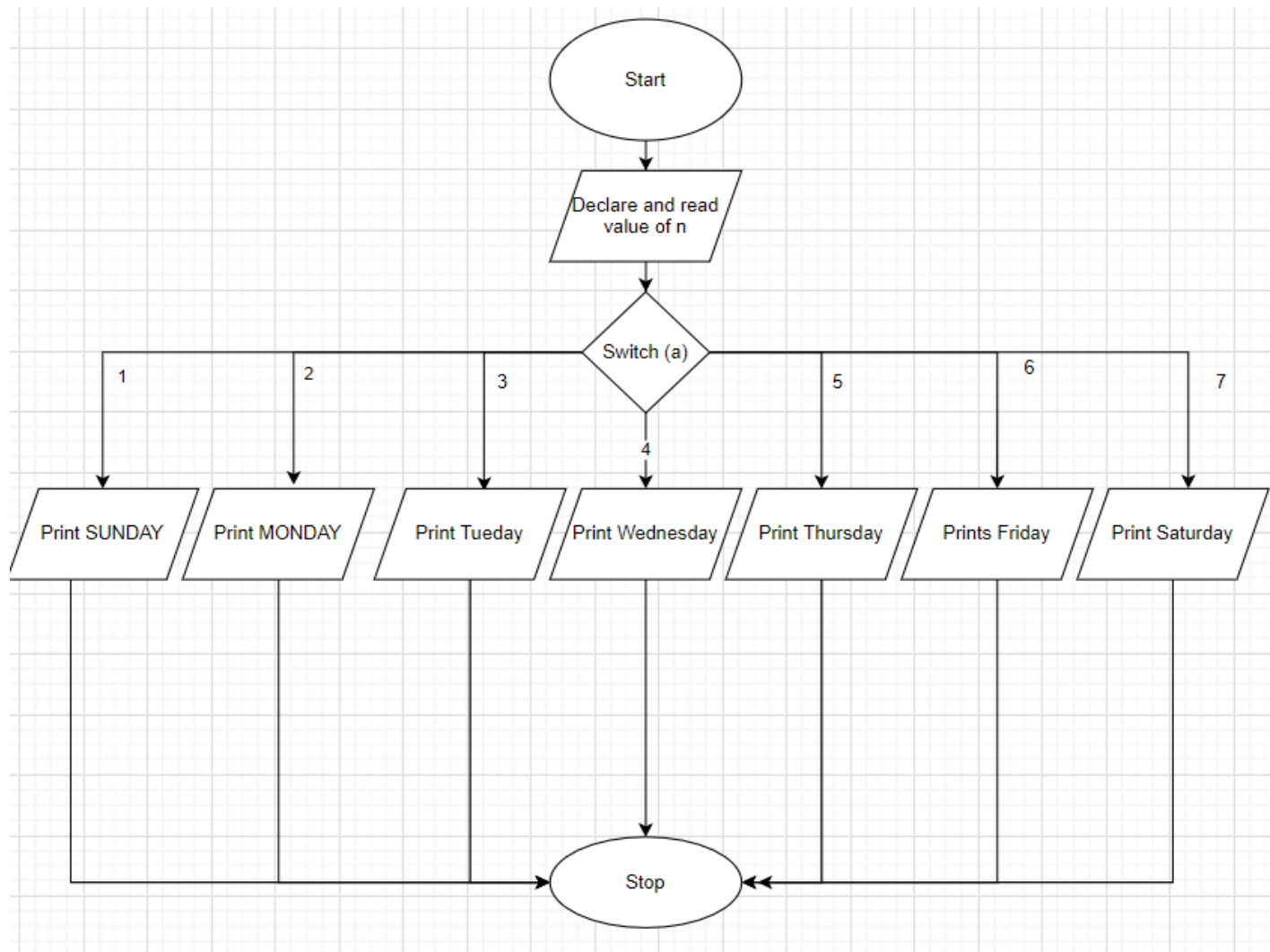
8. Write a program that prompts the user to input any integer from 1-7 and display the corresponding day in week.

Algorithm:

- Start
- Declare and read value of n
- Switch(n)
 - 1
 - Print SUNDAY
 - 2
 - Print MONDAY
 - 3
 - TUESDAY
 - 4

- WEDNESDAY
 - 5
 - THURSDAY
 - 6
 - FRIDAY
 - 7
 - SATURDAY
- Stop

Flow Chart:



Source Code:

```

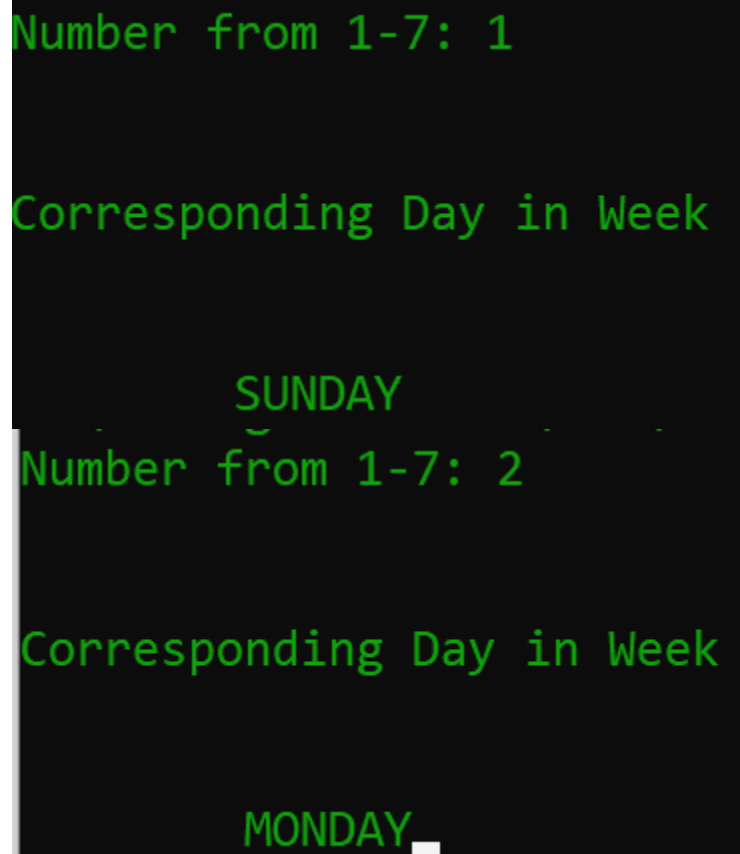
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

```

```
int main()
{
    int day_number;
    // system("cls");
    printf("Number from 1-7: ");
    scanf("%d", &day_number);
    printf("\n\nCorresponding Day in Week\n\n\n\t");
    switch (day_number)
    {
        case 1:
            printf("SUNDAY");
            break;
        case 2:
            printf("MONDAY");
            break;
        case 3:
            printf("TUESDAY");
            break;
        case 4:
            printf("WEDNESDAY");
            break;
        case 5:
            printf("THURSDAY");
            break;
        case 6:
            printf("FRIDAY");
            break;
        case 7:
```

```
printf("SATURDAY");  
break;  
default:  
printf("\t!!!ERROR!!!\n\n\tYou didn't give asked number.\n");  
break;  
}  
getch();  
return 0;  
}
```

Output:



```
Number from 1-7: 1  
  
Corresponding Day in Week  
  
SUNDAY  
Number from 1-7: 2  
  
Corresponding Day in Week  
  
MONDAY
```

Number from 1-7: 3

Corresponding Day in Week

TUESDAY

Number from 1-7: 4

Corresponding Day in Week

WEDNESDAY

Number from 1-7: 5

Corresponding Day in Week

THURSDAY

```
Number from 1-7: 6
```

```
Corresponding Day in Week
```

```
FRIDAY
```

```
Number from 1-7: 7
```

```
Corresponding Day in Week
```

```
SATURDAY
```

```
Number from 1-7: 78
```

```
Corresponding Day in Week
```

```
!!!ERROR!!!
```

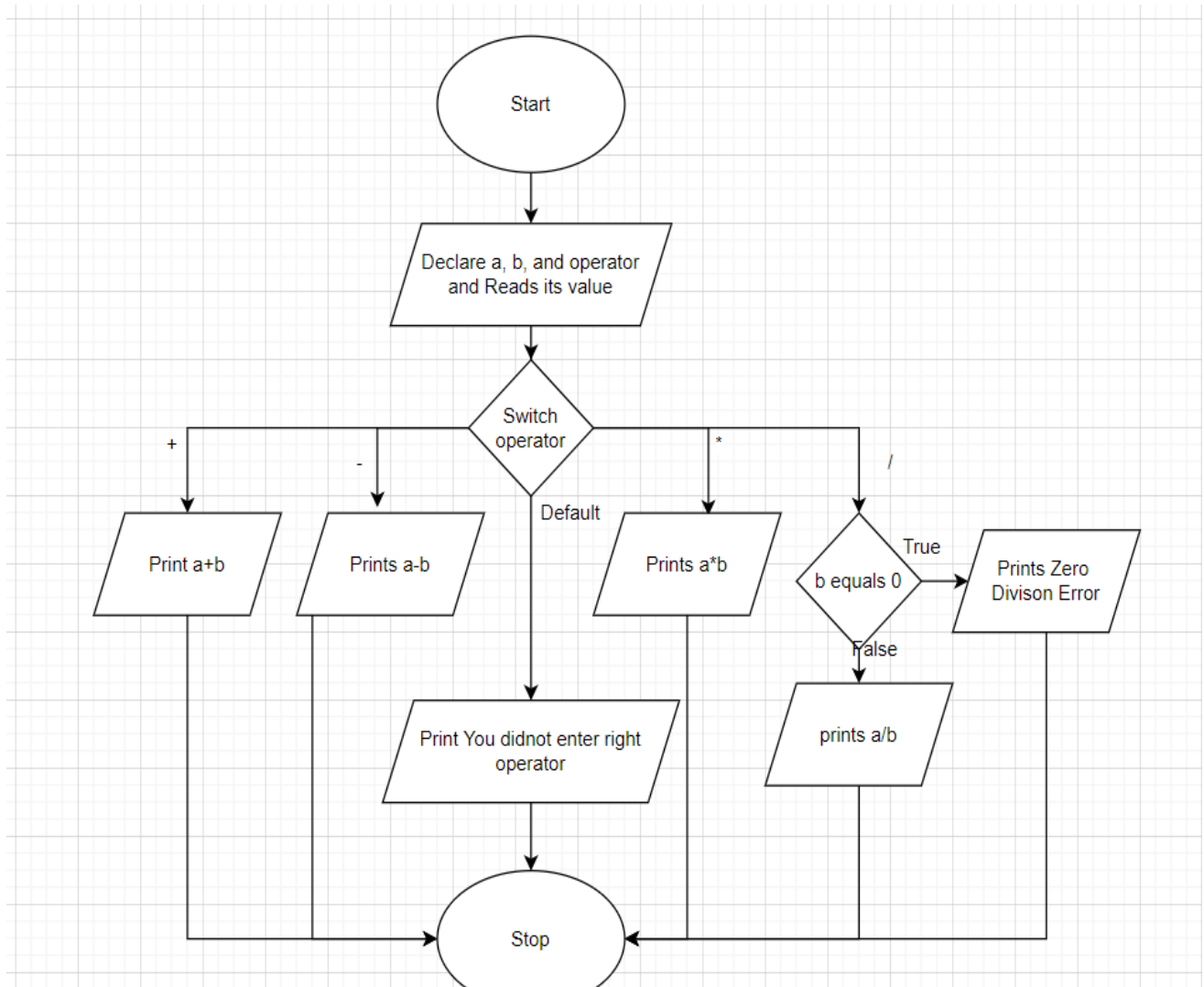
```
You didn't give asked number.
```

9. Write a program that ask the user a arithmetic operator and two operands and perform the corresponding operation

Algorithm:

- Start
- Declare a, b, and operator and read its value
- Switch(operator)
 - '+'
 - Print a+b
 - '-'
 - Print a-b
 - '*'
 - Print a*b
 - '/'
 - B equal to 0
 - True
 - Prints Zero division error
 - False:
 - Print a/b
 - Default
 - Print you didn't give correct operator
- Stop

Flow Chart:



Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    int operand_one, operand_two;
    char operator;
    float answer;
    system("cls");

```



```
printf("Operand one: ");
scanf("%d", &operand_one);
printf("Operand two: ");
scanf("%d", &operand_two);
printf("Operator: ");
scanf(" %c", &operator);
switch (operator)
{
case '+':
answer = operand_one + operand_two;
break;
case '-':
answer = operand_one - operand_two;
break;
case '*':
answer = operand_one * operand_two;
break;
case '/':
if (operand_two == 0)
{
printf("\nError! Zero Division error");
exit(0);
}
else
{
answer = (float)operand_one / operand_two;
break;
}
```

```
default:
printf("\nYou didn't give correct operator.");
exit(0);
}
printf("\nValue is %.2f.", answer);
getch();
return 0;
}
```

Output:

Operand one: 4 Operand two: 9 Operator: + Value is 13.00.	Operand one: 8 Operand two: 3 Operator: - Value is 5.00.
Operand one: 5 Operand two: 8 Operator: * Value is 40.00.	Operand one: 8 Operand two: 6 Operator: / Value is 1.33.
Operand one: 8 Operand two: 0 Operator: / Error! Zero Division error	

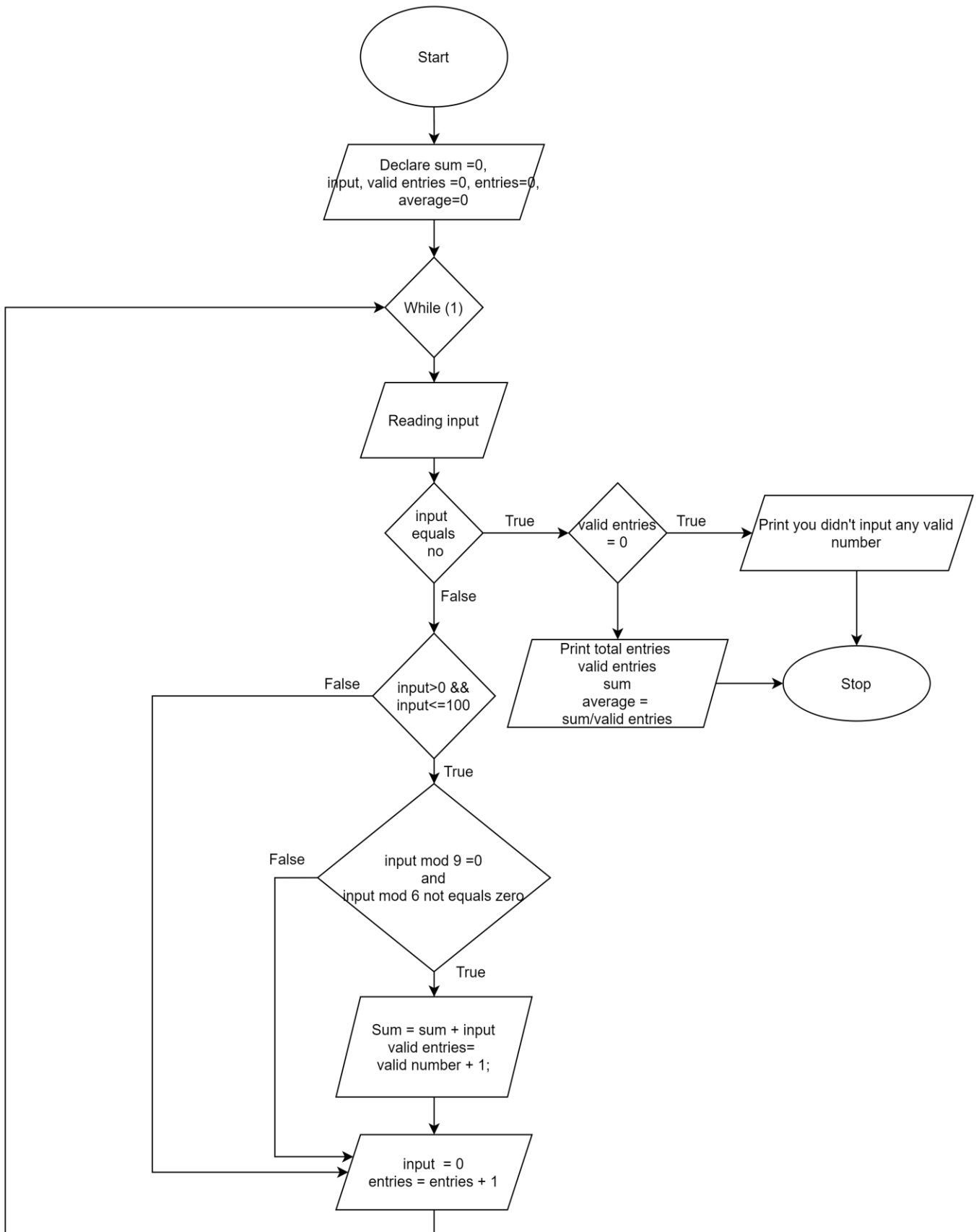
10."You are given a task to develop a system to read at least 100 integer numbers and continue until the user enters NO. Your system must have capacity to calculate the sum and average of those numbers which are exactly divisible by 9 but not by 6 and lies in between 1 to 100 and display a suitable message if no such number is read."

Write algorithm, flowchart and code to develop the system

Algorithm:

- Start
- Declare sum = 0, input, valid entries = 0, entries = 0, average = 0
- While(1)
 - Reads input
 - Input equals no
 - True
 - Valid equal zero
 - True
 - Print you did not enter any valid number.
 - Stop
 - False
 - Print Total entries, valid entries, sum, average =sum/valid entries
 - Stop
 - False
 - Input >0 and input <=100
 - True
 - Input mod 9 equals 0 and input mod 6 not equal 0
 - True
 - Sum = sum + input
 - Valid entries = valid entries +1
 - False:
 - Input = 0
 - Entries = entries +1

Flow Chart:



Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
    int sum = 0, input_number = 0, no_of_valid_number = 0, valid_number[500], entries = 1;
    float average = 0;
    char input[100];
    system("cls");
    printf("\t\t <-----QUESTION 10 SYSTEM----->\n\n\n");
    printf("\t\t <-----TAKING INPUT----->\n\n\n");
    while (1)
    {
        printf("\nValue %d : ", entries);
        scanf("%s", input);
        if ((strcmp(input, "No") == 0) || (strcmp(input, "no") == 0) || (strcmp(input, "NO") == 0))
        {
            if (no_of_valid_number == 0)
            {
                system("cls");
                printf("\t\t <-----QUESTION 10 SYSTEM----->\n\n\n");
                printf("\t\t <-----SYSTEM ANALYSIS RESULT----->\n\n\n");
                printf("\t\t!!!!!!ERROR!!!!!!\n\n");
                printf(" You didn't enter the number with following characteristic:\n\n");
            }
        }
    }
}
```

```

printf("\t\t-->Divisible by 9\n");
printf("\t\t-->Not divisible by 6\n");
printf("\t\t-->Number should be between 1 to 100.\n\n");
printf("\n\n\t\t<---Thanks for using our sytem--->");
getch();
exit(0);
}
else
{
system("cls");
printf("\t\t<-----QUESTION 10 SYSTEM----->\n");
printf("\t\t<-----SYSTEM ANALYSIS RESULT----->\n\n\n");
printf("Total number of entries: %d\n", entries - 1);
printf("Valid Entries: %d\n\n", no_of_valid_number);
printf("\nValid Entries: ");
for (int i = 0; i < no_of_valid_number; i++)
{
printf(" %d", valid_number[i]);
}
average = (float)sum / no_of_valid_number;
printf("\n\nSum : %d", sum);
printf("\nAverage = %.2f", average);
printf("\n\n\n\t\t<---Thanks for using our sytem--->");
getch();
exit(0);
}
}
else

```

```
{  
for (int i = 0; i < strlen(input); i++)  
{  
    char c = input[i];  
    if (c == '1')  
    {  
        input_number = input_number * 10 + 1;  
    }  
    else if (c == '2')  
    {  
        input_number = input_number * 10 + 2;  
    }  
    else if (c == '3')  
    {  
        input_number = input_number * 10 + 3;  
    }  
    else if (c == '4')  
    {  
        input_number = input_number * 10 + 4;  
    }  
    else if (c == '5')  
    {  
        input_number = input_number * 10 + 5;  
    }  
    else if (c == '6')  
    {  
        input_number = input_number * 10 + 6;  
    }  
}
```

```
else if (c == '7')
{
input_number = input_number * 10 + 7;
}
else if (c == '8')
{
input_number = input_number * 10 + 8;
}
else if (c == '9')
{
input_number = input_number * 10 + 9;
}
else if (c == '0')
{
input_number = input_number * 10 + 0;
}
}
// printf("%d", input_number);
}
if (input_number > 0 && input_number <= 100)
{
if ((input_number % 9 == 0) && (input_number % 6 != 0))
{
sum = input_number + sum;
valid_number[no_of_valid_number] = input_number;
no_of_valid_number++;
}
}
```



```
input_number = 0;
```

```
entries++;
```

```
}
```

```
}
```

Output:

```
<-----QUESTION 10 SYSTEM----->
```

```
<-----TAKING INPUT----->
```

```
Value 1 : 45
```

```
Value 2 : 89
```

```
Value 3 : 15
```

```
Value 4 : 93
```

```
Value 5 : 4651
```

```
Value 6 : 9
```

```
Value 7 : 42
```

```
Value 8 : 72
```

```
Value 9 : 81
```

```
Value 10 : 56
```

```
Value 11 : 99
```

```
Value 12 : 6
```

```
Value 13 : no
```

```
<-----QUESTION 10 SYSTEM----->
<-----SYSTEM ANALYSIS RESULT----->

Total number of entries: 12
Valid Entries: 4

Valid Entries: 45 9 81 99

Sum : 234
Average = 58.50

<---Thanks for using our sytem--->_
```

For no any valid numbers,

```
<-----QUESTION 10 SYSTEM----->

<-----SYSTEM ANALYSIS RESULT----->

!!!!!!!ERROR!!!!!!!

You didn't enter the number with following characteristic:

-->Divisible by 9
-->Not divisible by 6
-->Number should be between 1 to 100.

<---Thanks for using our sytem--->_
```

Analysis

Here, we have learned above various ways to taking input from user like **search set**(In Lab 3 program 11 and 12) for string, **no of element to take** (Lab 3 Program 10), and many more. In similar ways, we learn how to format our output like **how many characters to display, how any number to display after decimal, indentation gap** in output, **fixing the width** when display of variables (Lab 3 Program 4 5 6 7 8).

Moreover, we learn how the **if else statement, if else if ladder, for loop, while loop, do... while loop** works.

Also, it helps to understand the workings of these statements and building logic on decision taking program.

Conclusion

In nut shell, we became acquaintance with Formatted **input/ output** and **Looping**. Also, we shape ourselves with these newly learnt statements.

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 1 and 2

Title: **Repetitive Structure (Branching Looping)**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Repetitive Structure

Code and Output

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

In the Windows search bar, type 'settings' to open your Windows Settings.
Search for Edit environment variables for your account.
Choose the Path variable and then select Edit.
Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin**.
Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

Repetitive Structure

For Loop

A for loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

Syntax:

```
for ( init; condition; increment)
    {statement(s);}
```

Here is the flow of control in a 'for' loop –

- The init step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You do not have to put a statement here, as long as a semicolon appears.
- Next, the condition is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and the flow of control jumps to the next statement just after the 'for' loop.
- After the body of the 'for' loop executes, the flow of control jumps back up to the increment statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.

- The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again condition). After the condition becomes false, the 'for' loop stops.

While Loop

A while loop in C programming repeatedly executes a target statement as long as a given condition is true.

Syntax

```
while(condition)
{statement(s);}
```

Here, statement(s) may be a single statement or a block of statements. The condition may be any expression, and true is any nonzero value. The loop iterates while the condition is true.

When the condition becomes false, the program control passes to the line at once following the loop.

Do ...While Loop

A do...while loop is similar to a while loop, except the fact that it is guaranteed to execute at least one time.

Syntax:

```
do {statement(s);}
while (condition);
```

The conditional expression appears at the end of the loop, so the statement(s) in the loop executes once before the condition is tested.

If the condition is true, the flow of control jumps back up to do, and the statement(s) in the loop executes again. This process repeats until the given condition becomes false.

CODE AND OUTPUT

2. Write a program to read an unsigned integer (suppose n) and display from 1 to n and n to 1

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int n;
    system("cls");
    printf("Enter the value of n: ");
    scanf("%d", &n);
    printf("From 1 to %d",n);
    printf("\t\tFrom %d to 1", n);
    for (int i = 1; i <= n; i++)
    {
        printf("\n %3d\t\t%3d\n", i, (n - i + 1));
    }
    getch();
    return 0;
}
```

Output:

```
Enter the value of n : 5
From 1 to 5           From 5 to 1
  1                   5
  2                   4
  3                   3
  4                   2
  5                   1
```

3. Write a program to display the sum of even numbers from 1 to n (n is an unsigned integer)

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int main ()
```

```
{
```

```
    unsigned int n;
```

```
    int sum_of_even;
```

```
    system("cls");
```

```
    printf("Value of n: ");
```


```

scanf("%u", &n);
for (int i = 1; i <= n; i++)
{

    if((i % 2) == 0)
    {
        sum_of_even = sum_of_even + i;
    }
}
printf("\nSum of even number from 1 to %d is %d",n, sum_of_even);
getch();
return 0;
}

```

Output:



```

Value of n: 10

Sum of even number from 1 to 10 is 30

```

4. Write a program to read an integer and find out product from 1 to n if n is even and sum from 0 to n if n is odd.

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int n, product = 1, sum = 0;

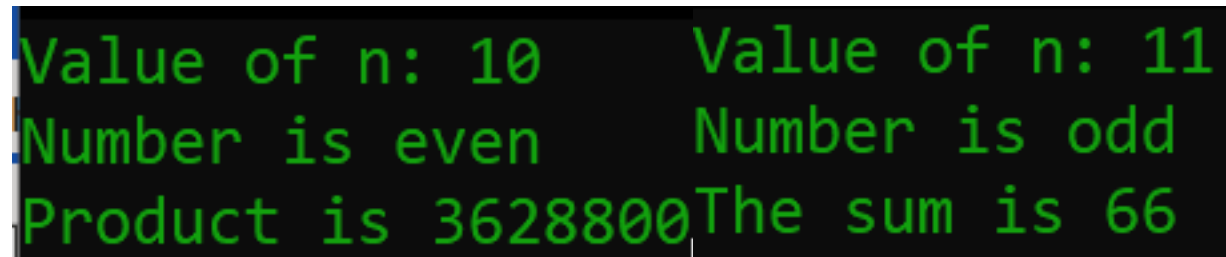
```

```

system("cls");
printf("Value of n: ");
scanf("%u", &n);
if(n%2==0) {
    printf("Number is even\n");
    for (int i = 1; i <= n;i++)
    {
        product = product * i;
    }
    printf("Product is %d", product);
}
else {
    printf("Number is odd\n");
    for (int i = 0; i <= n;i++) {
        sum = sum + i;
    }
    printf("The sum is %d", sum);
}
getch();
return 0;
}

```

Output:



```

Value of n: 10      Value of n: 11
Number is even      Number is odd
Product is 3628800  The sum is 66

```

5. Write a program to read an integer and compute its factorial and display proper message if its factorial cannot be completed

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

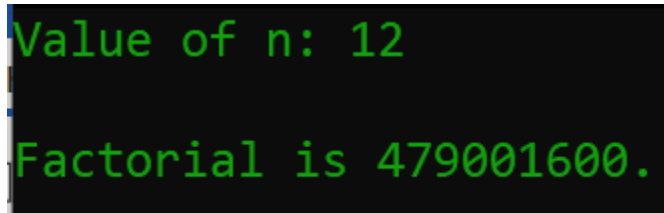
int main ()
{
    float n;
    int factorial = 1;
    system("cls");
    printf("Value of n: ");
    scanf("%f", &n);

    if ((n - (int)n) == ((float)2-2))
    {
        if (n > 0)
        {
            if (n == 0 || n == 1)
            {
                printf("\nFactorial is %d", 1);
            }
            else
            {
                for (int i = 1; i <= n; i++)
                {
```

```
        factorial = factorial * i;
    }
    printf("\nFactorial is %d.", factorial);
}
else
{
    printf("Expected: Positive Integer\nGiven: Negative Integer\n");
}
}
else
{
    printf("Expected: Integer\nGiven: floating number.");
}

    getch();
    return 0;
}
```

Output:

A screenshot of a terminal window with a black background and green text. The first line shows 'Value of n: 12' and the second line shows 'Factorial is 479001600.'.

```
Value of n: 12
Factorial is 479001600.
```

6. Write a program to calculate $x^n/n!$ where x is floating point number and n is integer

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main ()
{
    float x;
    int n, factorial=1;
    system("cls");
    printf("Value of x: ");
    scanf("%f", &x);
    printf("Value of n: ");
    scanf("%d", &n);
    for (int i = 1; i <= n; i++) {
        factorial = factorial * i;
    }
    printf("Value  $x^n / n!$  is %.3f.", (pow (x, n) / (double)factorial));
    getch();
    return 0;
}
```

Output:


```
Value of x: 5
Value of n: 2
Value x^n / n! is 12.500.
```

7. Write a program to display the fibonacci series

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int a = 0, b = 1, c;
    int n;

    system("cls");
    printf("Value of n: ");
    scanf("%d", &n);
    printf("Fibonacci Number: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", a);
        c = a + b;
```

```

    a = b;
    b = c;
}
    getch();
    return 0;
}

```

Output:

```

Value of n: 10
Fibonacci Number: 0 1 1 2 3 5 8 13 21 34

```

8. Write a program to read a positive number and find its sum of digits in it.

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int number, remainder, sum;

    system("cls");

    printf("Enter the positive integers: ");
    scanf("%d", &number);

    printf("Number: %d", number);
    while (number != 0)
    {
        remainder = number % 10;

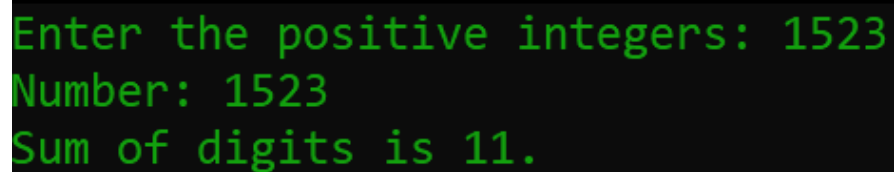
```

```

        sum = sum + remainder;
        number = number / 10;
    }
    printf("\nSum of digits is %d.", sum);
    getch();
    return 0;
}

```

Output:



```

Enter the positive integers: 1523
Number: 1523
Sum of digits is 11.

```

9.1] Write a program to read number and check whether it is palindrome or not

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int number, reverse, remainder, copy_number;
    system("cls");
    printf("Number: ");
    scanf("%d", &number);
    copy_number = number;
    while (copy_number != 0)
    {

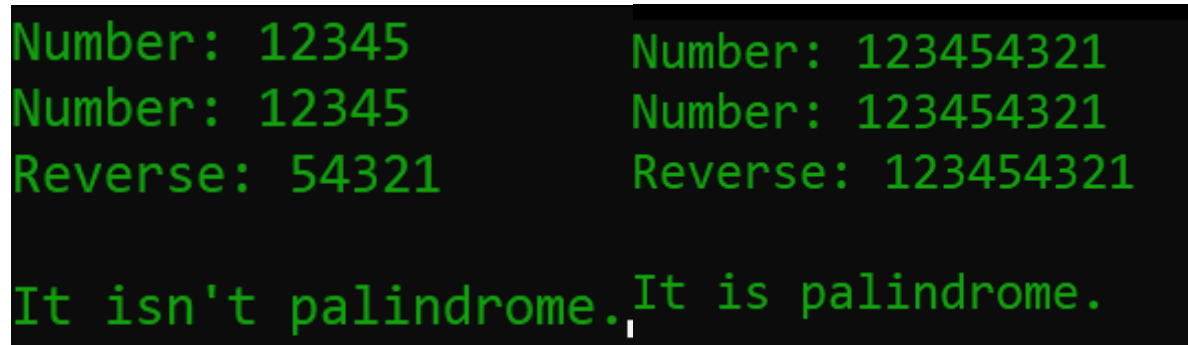
```

```

    remainder = copy_number % 10;
    reverse = reverse * 10 + remainder;
    copy_number = copy_number / 10;
}
printf("Number: %d\nReverse: %d\n\n", number, reverse);
if (number == reverse)
{
    printf("It is palindrome.");
}
else
{
    printf("It isn't palindrome.");
}
getch();
return 0;
}

```

Output:



The screenshot displays the output of the program for two different input numbers. The first test case uses the number 12345, resulting in a reverse of 54321 and the message "It isn't palindrome.". The second test case uses the number 123454321, resulting in a reverse of 123454321 and the message "It is palindrome.". The output is presented in a dark-themed window with green text.

```

Number: 12345          Number: 123454321
Number: 12345          Number: 123454321
Reverse: 54321         Reverse: 123454321

It isn't palindrome.  It is palindrome.

```

9.2] Write a program to read number and check whether it is Armstrong or not

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
#include <math.h>
```

```
int main ()
```

```
{
```

```
    int armstrong = 0, remainder, number, copy_number;
```

```
    system("cls");
```

```
    printf("Number: ");
```

```
    scanf("%d", &number);
```

```
    copy_number = number;
```

```
    while (copy_number != 0)
```

```
    {
```

```
        remainder = copy_number % 10;
```

```
        armstrong = pow (remainder, 3) + armstrong;
```

```
        copy_number = copy_number / 10;
```

```
    }
```

```
    printf("\n\nNumber: %d\nArmstrong: %d\n\n", number, armstrong);
```

```
    if (number == armstrong)
```

```
    {
```

```
        printf("It is armstrong.");
```

```
    }
```

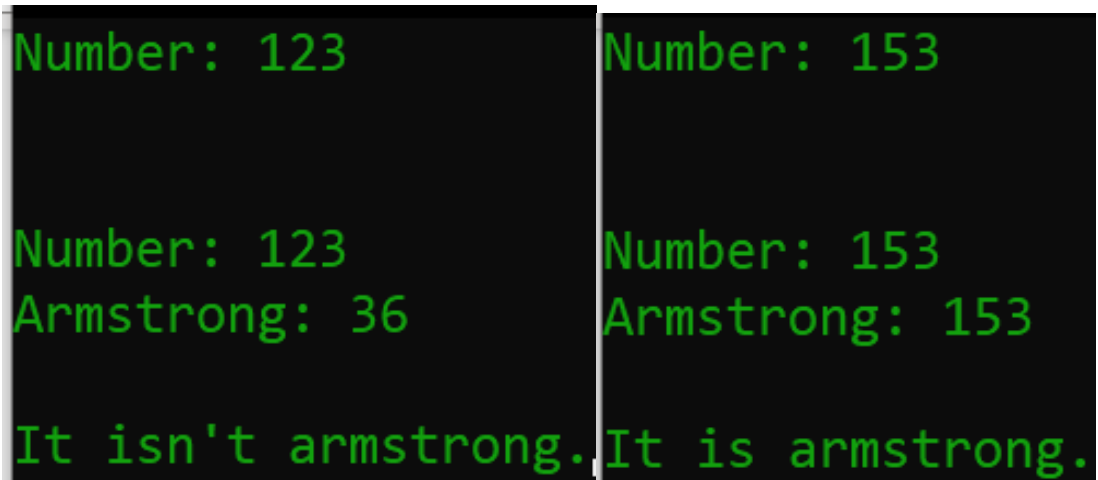
```
    else
```

```
    {
```

```
        printf("It isn't armstrong.");
```

```
}  
    getch();  
    return 0;  
}
```

Output:



The image shows two side-by-side screenshots of a terminal window with a black background and green text. The left screenshot shows the output for the number 123, and the right screenshot shows the output for the number 153. Both screenshots show the number, its Armstrong sum, and a message indicating whether it is an Armstrong number.

Number	Armstrong Sum	Result
123	36	It isn't armstrong.
153	153	It is armstrong.

9.3] Write a program to read number and check whether it is prime or not

Source Code:

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
  
int main ()  
{  
  
    int number, i;  
    system("cls");
```

```
printf("Enter the positive integer: ");
```

```
scanf("%d", &number);
```

```
i = number - 1;
```

```
if (number == 0 || number == 1)
```

```
{
```

```
    printf("It is neither prime nor composite.");
```

```
}
```

```
else
```

```
{
```

```
    for (; i; i--)
```

```
    {
```

```
        if (!(i == 0 || i == 1))
```

```
        {
```

```
            if (number % i == 0)
```

```
            {
```

```
                printf("Composite Number\n");
```

```
                break;
```

```
            }
```

```
        }
```

```
else
```

```
{
```

```
    printf("Prime Number");
```

```

    }
}

getch();
return 0;
}

```

Output:

```

Enter the positive integer: 156
Composite Number

```

```

Enter the positive integer: 97
Prime Number

```

9.4] Write a program to read two number and check whether it is twin prime or not

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int number1, number2;
    int twin_number1 = 1, twin_number2 = 1;

    float n1, n2;

```



```
system("cls");
```

```
printf("Number one: ");
```

```
scanf("%d", &number1);
```

```
printf("Number two: ");
```

```
scanf("%d", &number2);
```

```
if (number1 == 1 || number2 == 1 || number1 == 1 || number2 == 1)
```

```
{
```

```
    printf("\nIt isn't twin number.");
```

```
    getch();
```

```
    exit (0);
```

```
}
```

```
if ((number1 - number2) == 2 || (number2 - number1) == 2)
```

```
{
```

```
    for (int i = 2; i < number1; i++)
```

```
    {
```

```
        if (number1 % i == 0)
```

```
        {
```

```
            twin_number1 = 0;
```

```
            break;
```

```
        }
```

```
    }
```

```
    for (int i = 2; i < number2; i++)
```

```
    {
```

```
        if (number2 % i == 0)
```

```
        {
```

```

        twin_number2 = 0;
        break;
    }
}
if (twin_number1 == 1 && twin_number2 == 1)
{
    printf("\nIt is twin number");
}
else
{
    printf("\nIt isn't twin number.");
}
}
else
{
    printf("\nIt isn't twin prime number.");
}
getch();
return 0;
}

```

Output:

```
Number one: 7
Number two: 11

It isn't twin prime number.

Number one: 41
Number two: 43

It is twin number
```

10. Write a program to read a number (n) and display its multiplication table up to 10.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int number;
    system("cls");
    printf("Number: ");
    scanf("%d", &number);
    printf("Multiplication Table of %d\n\n", number);
    for (int i = 1; i <= 10; i++)
    {
        printf("%d * %d = %d\n", i, number, i * number);
    }
}
```

```
    getch();  
    return 0;  
}
```

Output:

```
Number: 12  
Multiplication Table of 12  
  
1   *   12   =   12  
2   *   12   =   24  
3   *   12   =   36  
4   *   12   =   48  
5   *   12   =   60  
6   *   12   =   72  
7   *   12   =   84  
8   *   12   =   96  
9   *   12   =  108  
10  *   12   =  120
```

Analysis

Here, we have learned above various ways iteration over the loop. We come across knowledge of **for loop**, **while loop**, **do...while loop**. We learnt in what kind of condition we should use this looping statement. We use for loop when we know the exact number of iterations and we use while or do...while loop when we do not know number of iterations.

Also, it helps to understand the workings of these statements and building logic on Looping and iterations taking program.

Conclusion

Conclusion

In nut shell, we became acquaintance with **Branching structure**. Also, we shape ourselves with these newly learnt statements.

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 1 and 2

Title: **Nested Looping Structure**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Nested Loop Structure

Code and Output

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

In the Windows search bar, type 'settings' to open your Windows Settings.

Search for Edit environment variables for your account.

Choose the Path variable and then select Edit.

Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**

Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

Nested Looping Structure

If a loop is present inside another loop then it is called nested loop structure

```
while(test_expression)
{
    statements;          while(test_expression)    {      statements;
} }
```

```
for ( init; condition; increment)
{
    for ( init; condition; increment)
    { statement(s); }
    statement(s);
}
do {
statement(s);
```

```
        do {    statement(s);  } while (condition );}
while (condition);
```

CODE AND OUTPUT

1.Program to print a multiplication table of MXN. Read the values of M and N from the user.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int m, n;
    system("cls");
    printf("Multiplication Table of: ");
    scanf("%d", &m);
    printf("Upto : ");
    scanf("%d", &n);
    printf("Multiplication Table of %d from 1 to %d is: \n", m, n);
    for (int i = 1; i <= n; i++)
    {
        printf("%d * %d = %d\n", m, i, m * i);
    }
    getch();
    return 0;
}
```

Output:

```
Multiplication Table of : 15
Upto : 13
Multiplication Table of 15 from 1 to 13 is :
15 * 1 = 15
15 * 2 = 30
15 * 3 = 45
15 * 4 = 60
15 * 5 = 75
15 * 6 = 90
15 * 7 = 105
15 * 8 = 120
15 * 9 = 135
15 * 10 = 150
15 * 11 = 165
15 * 12 = 180
15 * 13 = 195
```

2. Write a program to display the chessboard pattern.

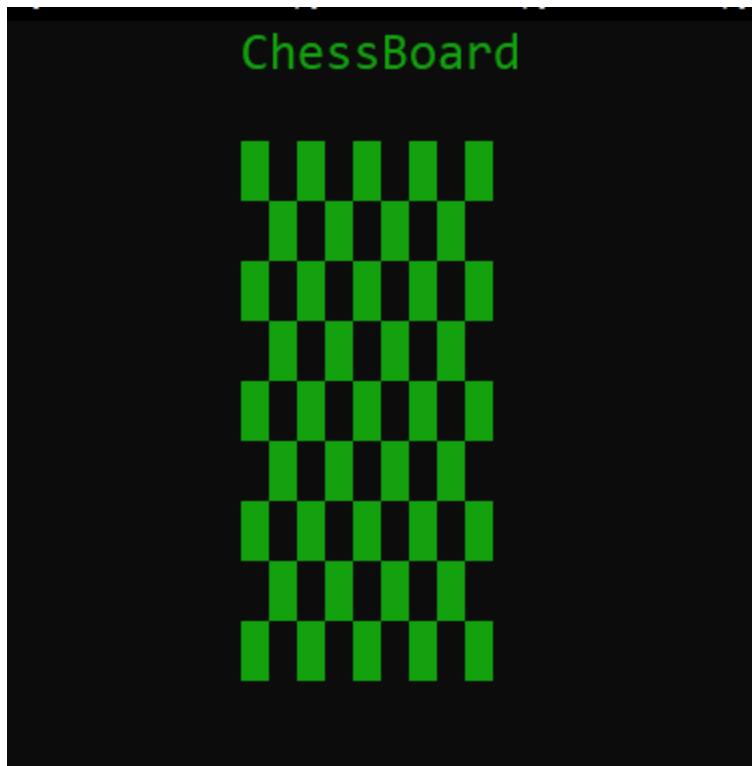
Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    system("cls");
    printf("\t\tChessBoard\n\n");
    for (int i = 0; i < 9; i++)
    {
        printf("\t\t");
        for (int j = 0; j < 9; j++)
```

```
{
    if ((i + j) % 2 == 0)
    {
        printf("\xdb");
    }
    else
    {
        printf(" ");
    }
}
printf("\n");
}
getch();
return 0;
}
```

Output:



3. Write a program to read two integer (n1 and n2, both positive and $n1 < n2$) from the user and display the prime and palidrome number between n1 and n2. Display their count also

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    unsigned int n1, n2, prime_count = 0, remainder, reverse = 0, copy_number,
    pallidrome_count = 0, number;

    system("cls");
    printf("Value of n1: ");
    scanf("%u", &n1);
    printf("Value of n2: ");
```

```
scanf("%u", &n2);  
for (int number = n1; number <= n2; number++)  
{  
    int i = number - 1;  
    if (number == 0 || number == 1)  
    {  
    }  
    else  
    {  
        for (; i--)  
        {  
            if (!(i == 0 || i == 1))  
            {  
                if (number % i == 0)  
                {  
                    break;  
                }  
            }  
        }  
        else  
        {  
            prime_count++;  
        }  
    }  
}  
copy_number = number;  
while (copy_number != 0)  
{
```

```

        remainder = copy_number % 10;
        reverse = reverse * 10 + remainder;
        copy_number = copy_number / 10;
    }
    if (number == reverse)
    {
        pallidrome_count++;
    }
    else
    {
    }
    reverse = 0;
}

printf("\nFrom %d to %d\n", n1, n2);
printf("Number of primes is %d\n", prime_count);
printf("Number of paliidrome number is %d\n", pallidrome_count);
getch();
return 0;
}

```

Output:

```

Value of n1: 3
Value of n2: 15

From 3 to 15
Number of prime is 5
Number of paliidrome number is 8

```


4. Write a program to find the sum of all positive number entered by the user. Read the numbers and keep calculating the sum until the user enter 0. */

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main ()
{
    int number, sum = 0, count = 0;

    system("cls");

    while (1)
    {
        printf("Positive Number: ");

        scanf("%d", &number);

        if (number == 0)
        {
            break;
        }

        else if (number > 0)
        {
            sum = sum + number;

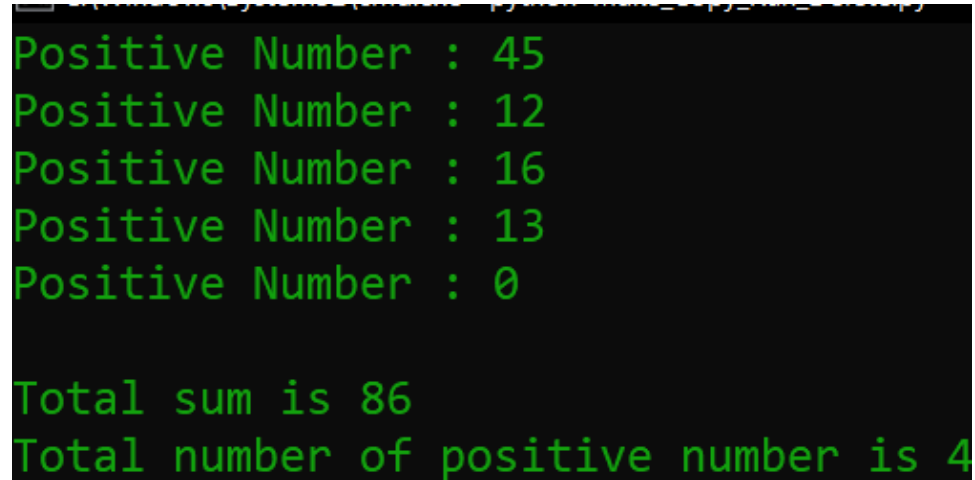
            count++;
        }
    }

    printf("\nTotal sum is %d\n", sum);

    printf("Total number of positive numbers is %d", count);
```

```
    getch();  
    return 0;  
}
```

Output:



```
Positive Number : 45  
Positive Number : 12  
Positive Number : 16  
Positive Number : 13  
Positive Number : 0  
  
Total sum is 86  
Total number of positive number is 4
```

5.1

Question:

1,2,3,4,5, 6.....n

Source Code:

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
  
int main ()  
{  
    int n;  
    system("cls");  
    printf("Value of n: ");  
    scanf("%d", &n);  
    printf("\nNumber for 1 to n is listed below: \n\n", n);
```

```

        for (int i = 1; i <= n; i++)
        {
            printf("%d ", i);
        }
        getch();
        return 0;
    }

```

Output:

```

Number for 1 to n is listed below:

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45
46 47 48 49 50

```

5.2

Questions:

2,4,6,8,102n

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    int n;
    system("cls");

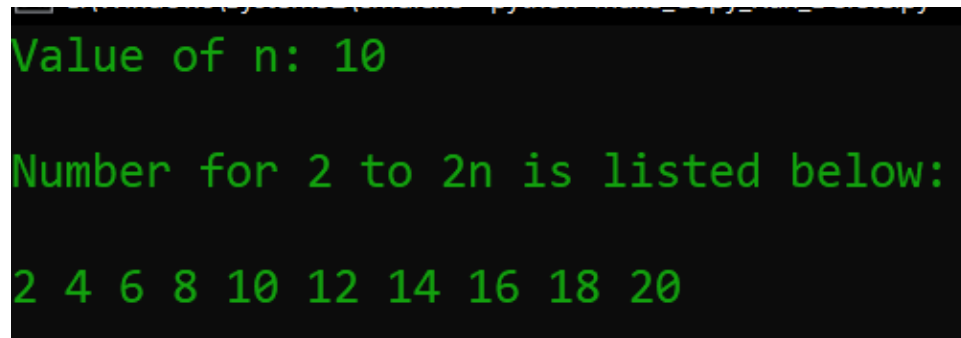
```

```

printf("Value of n: ");
scanf("%d", &n);
printf("\nNumber for 2 to 2n is listed below: \n\n", n);
for (int i = 1; i <= n; i++)
{
    printf("%d ", 2 * i);
}
getch();
return 0;
}

```

Output:



```

Value of n: 10

Number for 2 to 2n is listed below:

2 4 6 8 10 12 14 16 18 20

```

5.3

Question:

1,2,5,10,17,26.....

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    int n, term = 1;

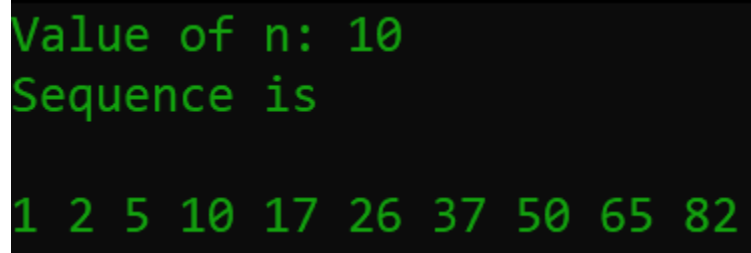
```

```

system("cls");
printf("Value of n: ");
scanf("%d", &n);
printf("Sequence is \n\n");
for (int i = 0; i < n; i++)
{
    printf("%d ", term);
    term = term + (2 * i + 1);
}
getch();
return 0;
}

```

Output:



```

Value of n: 10
Sequence is

1 2 5 10 17 26 37 50 65 82

```

5.4

$(1^2 + 2^2)/2, (2^2 + 3^2)/3, \dots$

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
int main ()

```

```

{
    int n;
    system("cls");
    printf("Value of n: ");
    scanf("%d", &n);
    printf("\nSequence is:\n");
    for (int i = 1; i <= n; i++)
    {
        printf("(%d^2 + %d^2)/%d ", i, i + 1, i + 1);
    }

    printf("\n\nRespective Value is:\n");
    for (int i = 1; i <= n; i++)
    {
        printf("%.3f ", (pow(i, 2) + pow(i + 1, 2)) / (i + 1));
    }

    getch();
    return 0;
}

```

Output:

```

Value of n: 8

Sequence is :
(1^2 + 2^2)/2  (2^2 + 3^2)/3  (3^2 + 4^2)/4  (4^2 + 5^2)/5
(5^2 + 6^2)/6  (6^2 + 7^2)/7  (7^2 + 8^2)/8  (8^2 + 9^2)/9

Respective Value is :
2.500 4.333 6.250 8.200 10.167 12.143 14.125 16.111

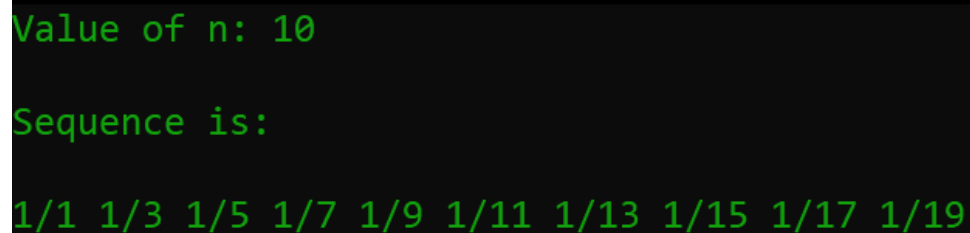
```

1,1/3,1/5.....,1/(2n-1)

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    int n;
    system("cls");
    printf("Value of n: ");
    scanf("%d", &n);
    printf("\nSequence is: \n\n", n);
    for (int i = 1; i <= n; i++)
    {
        printf("1/%d ", 2 * i - 1);
    }
    getch();
    return 0;
}
```

Output:

A screenshot of a terminal window with a black background and green text. The output shows the program's execution for n=10. It first displays 'Value of n: 10', then 'Sequence is:' on a new line, and finally the sequence of terms '1/1 1/3 1/5 1/7 1/9 1/11 1/13 1/15 1/17 1/19' on another line.

Value of n: 10

Sequence is:

1/1 1/3 1/5 1/7 1/9 1/11 1/13 1/15 1/17 1/19

6.1

Question:

2+4+6+8+10+.....+ 2n

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int n;
    system("cls");/*CLear the screen*/
    printf("Value of n: ");
    scanf("%d", &n);
    printf("\nSequence is: \n\n");
    for (int i = 1; i <= n; i++)
    {
        printf("%d ", 2 * i);
        if (n == i)
        {
            break;
        }
        else
        {
            printf("+ ");
        }
    }
    getch();
    return 0;
```



```
}
```

Output:

```
Value of n: 10
Sequence is:
2 + 4 + 6 + 8 + 10 + 12 + 14 + 16 + 18 + 20
```

6.2

Question:

$1 - 1/1! + 1/2! - 1/3! + \dots + (-1)^n / (n-1)! \quad n = 0, 1, 2, 3, \dots$

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main ()
{
    int n;
    system("cls");
    printf("Value of n: ");
    scanf("%d", &n);
    printf("\n\nSequence is:\n");
    for (int i = 1; i <= n; i++)
    {
        if (i == 1)
```

```

{
    printf("1 ");
}
else
{
    printf("1/(%d)! ", (i - 1));
}
if (n == i + 1)
{
    break;
}
else
{
    if (i % 2 != 0)
    {
        printf(" + ");
    }
    else
    {
        printf(" - ");
    }
}
}

printf("\n\nRespective Value:\n"); /* Information*/
for (int i = 0; i < n; i++)
{
    printf("%f ", ((float)1 / tgamma(i+1))); /*Print the value

```

```

    if (n == i+1)
    {
        break;
    }
    else
    {
        if (i%2 != 0)
        {
            printf(" + ");
        }
        else
        {
            printf(" - ");
        }
    }
}
getch();
return 0;
}

```

Output:

```

Sequence is :
1 + 1/(1)! - 1/(2)! + 1/(3)! - 1/(4)! + 1/(5)! - 1/(6)!

Respective Value :
1.000000 - 1.000000 + 0.500000 - 0.166667 + 0.041667 - 0.008333
+ 0.001389 - 0.000198

```

6.3

$$1 - x^2/2! + x^4/4! - \dots + (-1)^i x^{2i}/(2i)! \quad i = 0, 1, 2, 3, \dots$$

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main ()
{
    int x, n;

    system("cls");

    printf("Value of x: ");
    scanf("%d", &x);

    printf("Value of n: ");
    scanf("%d", &n);

    printf("\nSequence is:\n\n");

    for (int j = 0; j < n; j++)
    {
        if (j == 0)
        {
            printf("1 ");
        }
        else
        {
            printf("x^%d/(%d)! ", 2 * j, 2 * j);
        }

        if (n == j + 1)
        {
            break;
        }
    }
}
```

```

    }
    else {
        if (pow (-1, j) > 0)
        {
            printf(" - ");
        }
        else
        {
            printf(" + ");
        }
    }
}
printf("\n\nRespective Value:\n");

```

```

for (int k = 0; k < n; k++)
{
    if (k == 0)
    {
        printf("1 ");
    }
    else
    {
        printf("%.8f", pow (x, 2 * k) / tgamma(2 * k + 1));
    }
    If (n == k + 1)
    {
        break;
    }
}

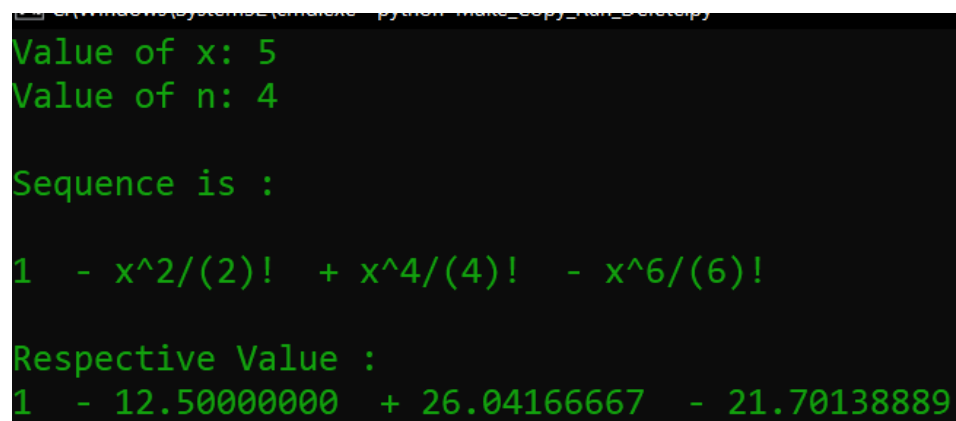
```

```

    }
    else
    {
        if (pow (-1, k) > 0)
        {
            printf(" - ");
        }
        else
        {
            printf(" + ");
        }
    }
}
}
getch();
return 0;
}

```

Output:



```

Value of x: 5
Value of n: 4

Sequence is :

1 - x^2/(2)! + x^4/(4)! - x^6/(6)!

Respective Value :
1 - 12.50000000 + 26.04166667 - 21.70138889

```

7.

Questions:

$1 + x/1! + x^2/2! + x^3/3! + \dots$ till sum of terms is less than 10^{-6}

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main ()
{
    int power = 0;
    double x, term = 1, sum_of_term = 0, condition;
    system("cls");
    printf("Value of x: ");
    scanf("%lf", &x);
    printf("\n\nSequence with value of x till term > 10^-6: \n");
    while (condition > (double)0) {
        term = pow(x, power) / tgamma(power + 1);
        printf("%.10f +", term);
        sum_of_term = term + sum_of_term;
        condition = term - 0.000001;
        power++;
    }
    printf("\n\nSum of term: %.5f\n", sum_of_term);
    printf("\n\nAt %d term, term is less than 10^-6\n", power);
    getch();
    return 0;
}
```

```
}
```

Output:

```
Value of x: 1

Sequence with value of x till term>10^-6:
1.0000000000 +1.0000000000 +0.5000000000 +0.1666666667 +0.0416666667
+0.0083333333 +0.0013888889 +0.0001984127 +0.0000248016 +0.00000275
57 +0.0000002756 +

Sum of term : 2.71828

At 11 term,term is less than 10^-6
```

8.1

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int main ()
```

```
{
```

```
    int n;
```

```
    system("cls");
```

```
    printf("Value of n: ");
```

```
    scanf("%d", &n);
```

```
    printf("Sequence is:\n\n");
```

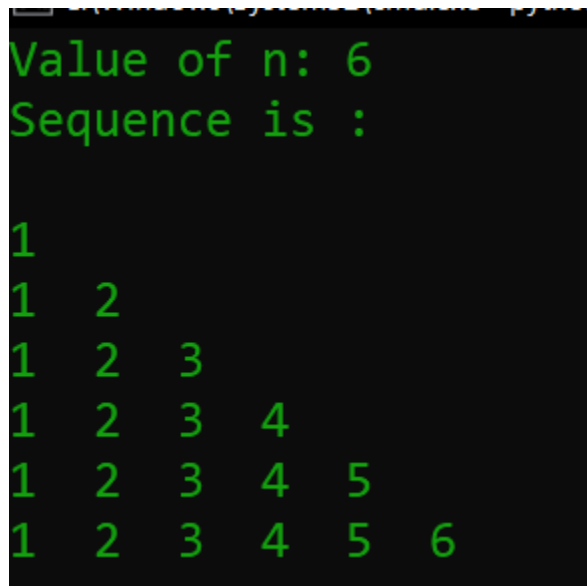


```

for (int i = 1; i <= n; i++)
{
    for (int j = 1; j <= i; j++)
    {
        printf("%d ", j);
    }
    printf("\n");
}
getch();
return 0;
}

```

Output:



```

Value of n: 6
Sequence is :

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6

```

8.2

5 4 3 2 1

5 4 3 2

5 4 3

5 4

5

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    int n;
    system("cls");
    printf("Value of n: ");
    scanf("%d", &n);
    printf("Sequence is:\n\n");
    for (int i = 0; i < n; i++)
    {
        for (int j = n; j > i; j--)
        {
            printf("%d ", j);
        }
        printf("\n");
    }
    getch();
    return 0;
}
```

Output:

```
Value of n: 5
Sequence is :
```

```
5  4  3  2  1
5  4  3  2
5  4  3
5  4
5
```

8.3

N

E E

P P P

A A A A

L L L L L

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int main ()
```

```
{
```

```
    system("cls");
```

```
    printf("Sequence is:\n");
```

```
    for (int i = 1; i <= 5; i++)
```

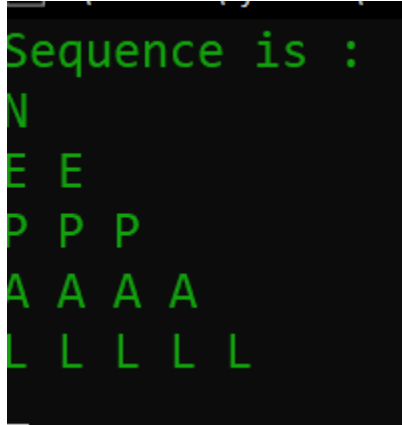
```
    {
```

```
for (int j = 0; j < i; j++)  
{  
    if (i == 1)  
    {  
        printf("N ");  
    }  
    else if (i == 2)  
    {  
        printf("E ");  
    }  
    else if (i == 3)  
    {  
        printf("P ");  
    }  
    else if (i == 4)  
    {  
        printf("A ");  
    }  
    else if (i == 5)  
    {  
        printf("L ");  
    }  
}  
printf("\n");  
}
```

```
getch();
```

```
    return 0;
}
```

Output:



```
Sequence is :
N
E E
P P P
A A A A
L L L L L
```

8.4

A

A B

A b C

A B C D

A b C d E

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    system("cls");
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j <= i; j++)
        {
```

```

    if (i % 2 == 0 && j % 2 != 0)
    {
        printf("%c ", 65 + j + 32);
    }
    else
    {
        printf("%c ", 65 + j);
    }
}
printf("\n");
}

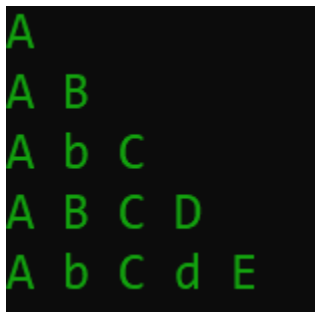
```

```

    getch();
    return 0;
}

```

Output:



```

A
A B
A b C
A B C D
A b C d E

```

8.5

Pattern:

```

#####
####**
###***
##****
#*****

```

* * * * *

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{

    system("cls");
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if(i+j<4) {
                printf("# ");
            }
            else {
                printf("* ");
            }
        }
        printf("\n");
    }
    getch();
    return 0;
}
```

Output:



8.6

```

      4
    3 4
  2 3 4
1 2 3 4
  2 3 4
    3 4
      4

```

Source Code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main ()
{
    system("cls");
    for (int i = 0; i < 7; i++)
    {
        for (int j = 1; j <= 4; j++)
        {
            if (i + j >= 4 && i - j != 3 && i - j != 4 && i - j != 5)

```



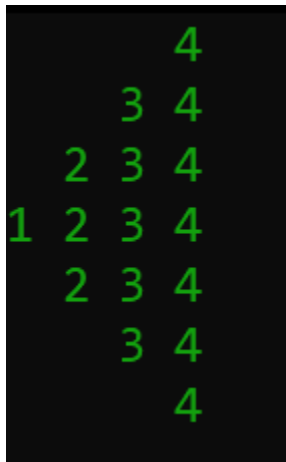
```

        {
            printf("%d ", j);
        }
        else
        {
            printf(" ");
        }
    }
    printf("\n");
}

getch();
return 0;
}

```

Output:



```

      2 3 4
    1 2 3 4
      2 3 4
        3 4

```

```

/*
*****
****
***

```

* *

*

*/

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    int n, space;
    system("cls");
    printf("Value of n: ");
    scanf("%d", &n);
    printf("Sequence is: \n");
    for (int i = n; i >= 0; i--)
    {
        space = n - i;
        while (space != 0)
        {
            printf(" ");
            space--;
        }
        for (int j = 0; j < i; j++)
        {
            printf("* ");
        }
    }
```

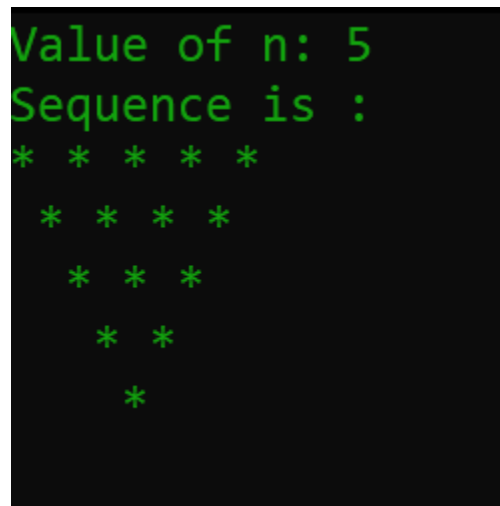
```

        printf("\n");
    }

    getch();
    return 0;
}

```

Output:



```

Value of n: 5
Sequence is :
* * * * *
 * * * *
  * * *
   * *
    *

```

8.8

Print the following pattern:

```

      0
    1  1
  2    2
3      3
2  4    4  2
1    5  5    1
0      6      0
1    5  5    1
  2  4    6  2

```

3 3
2 2
1 1
0

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    system("cls");
    for (int i = 0; i < 13; i++)
    {
        for (int j = 0; j < 13; j++)
        {
            if (i <= 6)
            {
                if ((i + j) == 6 || (j - i) == 6)
                {
                    if (i <= 3)
                    {
                        printf("%2d ", i);
                    }
                    else
                    {
                        if ((i + j) == 6) {
                            printf("%2d ", j);
                        }
                    }
                }
            }
        }
    }
}
```

```

    }
    else {
        printf("%2d", 12 - j);
    }
}
}
else if (i > 3 && (i == j || (i + j) == 12))
{
    printf("%2d ", i);
}
else
{
    printf(" ");
}
}
else if (i > 6)
{
    if (((i - j) == 6 || (i + j) == 18))
    {
        if (i <= 9)
        {
            if (i - j == 6) {
                printf("%2d ", j);
            }
            else {
                printf("%2d ", 12 - j);
            }
        }
    }
}

```

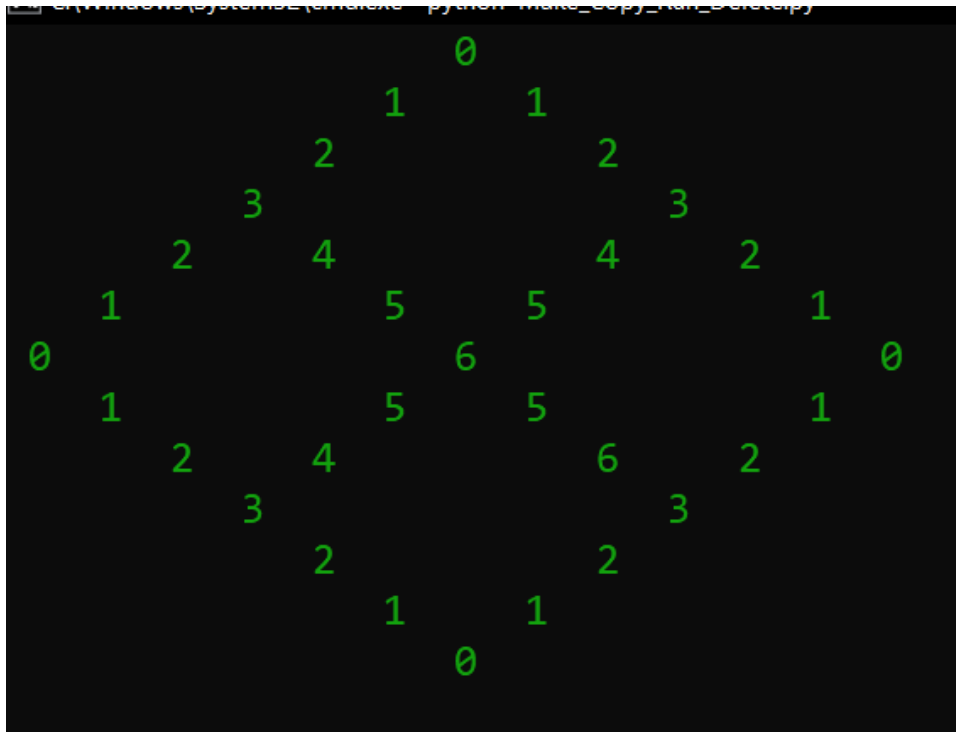
```

    }
    else
    {
        printf("%2d ", 12 - i);
    }
}
else if (i < 10 && (i == j || (i + j) == 12))
{
    if(i+j == 12) {
        printf("%2d ", j);
    }
    else {
        printf("%2d ", j - 2);
    }
}
else
{
    printf(" ");
}
}
}

printf("\n");
}
getch();
return 0;
}

```

Output:



8.9

```

      1
     1 2 1
    1 2 3 2 1
   1 2 3 4 3 2 1

```

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main ()
{
    system("cls");
    for (int i = 1; i <= 4; i++)

```

```

{
    for (int j = 0; j < 7; j++)
    {
        if (i + j >= 4 && j - i != 3 && j - i != 4 && j - i != 5)
        {
            if (j == 3)
            {
                printf(" %d  ", i);
            }
            else if (j == 2 || j == 4)
            {
                printf(" %d  ", i - 1);
            }
            else if (j == 1 || j == 5)
            {
                printf(" %d  ", i - 2);
            }
            else
            {
                printf(" %d  ", i - 3);
            }
        }
        else
        {
            printf("    ");
        }
    }
}

```



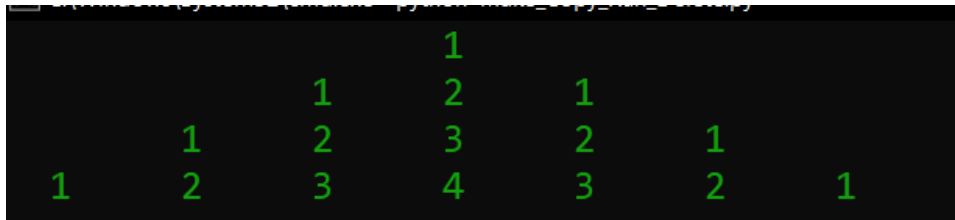
```

        printf("\n");
    }

    getch();
    return 0;
}

```

Output:



9. Write a program to find the sum of all positive number entered by the user. Read the numbers and keep calculating the sum until the user enter n.

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main ()
{
    int number,sum = 0, count = 0;
    char input_number;
    system("cls");
    while (1)
    {
        printf("Positive Number: ");
        scanf("%[n,0-9]", &input_number);
        if (input_number == 'n')

```

```

    {
        break;
    }
    else
    {
        number = input_number - 48;
        sum = sum + number;
        count++;
    }
}
printf("\nTotal sum is %d\n", sum);
printf("Total number of positive numbers is %d", count);
getch();
return 0;
}

```

Output:

```

Positive Number : 15
Positive Number : 16
Positive Number : 45
Positive Number : 16
Positive Number : 62
Positive Number : n

Total sum is 13
Total number of positive number is 5

```

Analysis

Through this lab activities we are able to understand concept behind nested loop

Why should we use it?

When we should use it what kind of program can be done using it?

Also, it builds our logic development in nested looping

Conclusion

We learn about nested loop.

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 7

Title: **Functions**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

C User Defined Function

Code and Output

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

- In the Windows search bar, type 'settings' to open your Windows Settings.
- Search for Edit environment variables for your account.
- Choose the Path variable and then select Edit.
- Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
- Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

C User Defined Function

A function is a block of code that performs a specific task.

C allows you to define functions according to your need. These functions are known as user-defined functions. For example:

Suppose, you need to create a circle and color it depending upon the radius and color. You can create two functions to solve this problem:

createCircle() function

color() function

Function prototype

A function prototype is simply the declaration of a function that specifies function's name, parameters and return type. It doesn't contain function body.

A function prototype gives information to the compiler that the function may later be used in the program.

Syntax of function prototype

returnType functionName(type1 argument1, type2 argument2, ...);

The function prototype is not needed if the user-defined function is defined before the main() function.

Calling a function

Control of the program is transferred to the user-defined function by calling it.

Syntax of function call

```
functionName(argument1, argument2, ...);
```

Function definition

Function definition contains the block of code to perform a specific task. In our example, adding two numbers and returning it.

Syntax of function definition

```
returnType functionName(type1 argument1, type2 argument2, ...){ //body of the function}
```

When a function is called, the control of the program is transferred to the function definition. And, the compiler starts executing the codes inside the body of a function.

1. Write a program to create a function float add(int ,float). The task of this function is to calculate the sum of passed value and return it to the calling function. Call this function from main() and display result

Source Code:

```
/*Header Files*/
```

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
/*Function defination*/
```

```
float add(int, float);
```

```
int main()
```



```

{
    /*Variable Declaration*/

    int number_one;

    float number_two, sum;


    system("cls"); /*Clear the Screen*/


    /*Taking input*/

    printf("Number one:");
    scanf("%d", &number_one);
    printf("Number two:");
    scanf("%f", &number_two);


    /*Function call*/

    sum = add(number_one, number_two);


    /*Desplay the result*/

    printf("\nAddition of %d and %.2f is %.2f", number_one, number_two, sum);


    getch(); /*Waits till a character is pressed*/
    return 0;
}

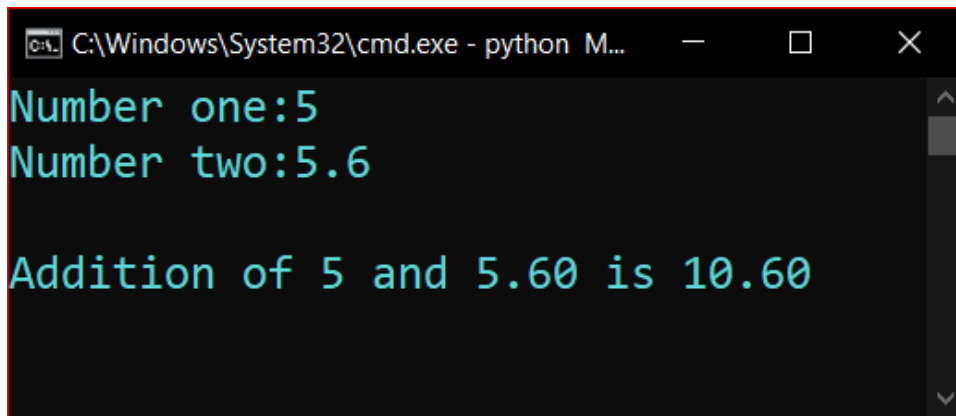

/*Function body*/

float add(int a, float b)
{
    return (float)a + b;
}

```

```
}
```

Output:



```
C:\Windows\System32\cmd.exe - python M...  
Number one:5  
Number two:5.6  
  
Addition of 5 and 5.60 is 10.60
```

2. Write a program to create a function void sumOfDigits(int);. This function must calculate the sum of digits in the given number and displays the sum.

Source Code:

```
#include <stdio.h>  
  
#include <conio.h>  
  
#include <stdlib.h>  
  
  
/*Function Defination*/  
int sumOfDigits(int);  
  
int main()  
{  
    /*Variable Declaration*/  
    int number, sum;  
  
    system("cls");/*Clear the screen*/  
  
    /*Taking input*/
```

```
printf("Number: ");  
scanf("%d", &number);
```

```
sum = sumOfDigits(number);
```

```
printf("Sum of digits in %d is %d", number, sum);
```

```
getch();
```

```
return 0;
```

```
}
```

```
/*Function Body*/
```

```
int sumOfDigits(int a)
```

```
{
```

```
    if (a < 0)
```

```
    {
```

```
        return 0;
```

```
    }
```

```
    else
```

```
    {
```

```
        while (a != 0)
```

```
        {
```

```
            int rem = a % 10;
```

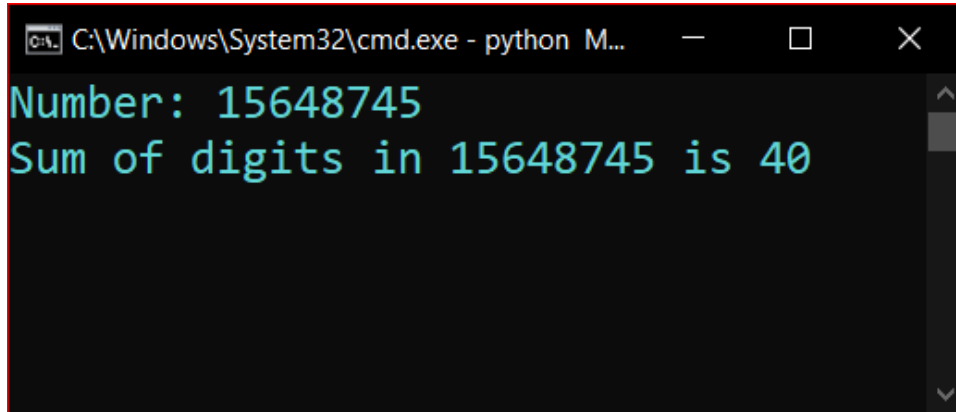
```
            a = a / 10;
```

```
            return rem + sumOfDigits(a);
```

```
        }
```

```
}  
}
```

Output:



```
C:\Windows\System32\cmd.exe - python M...  
Number: 15648745  
Sum of digits in 15648745 is 40
```

3. Write a program to read a non negative integer in main(). Pass this integer to a function fact() having return type unsigned integer. The function calculate the factorial of the received number and return to main() to display it.

Source Code:

```
#include <stdio.h>  
  
#include <conio.h>  
  
#include <stdlib.h>  
  
  
/*Function definition and body*/  
unsigned int fact(unsigned int a)  
{  
    if (a == 0)  
    {  
        return 1;  
    }  
    return a * fact(a - 1);  
}
```

```
int main()
{
    /*Variable Declaration*/
    unsigned number, factorial;

    system("cls");/*Clear the screen*/

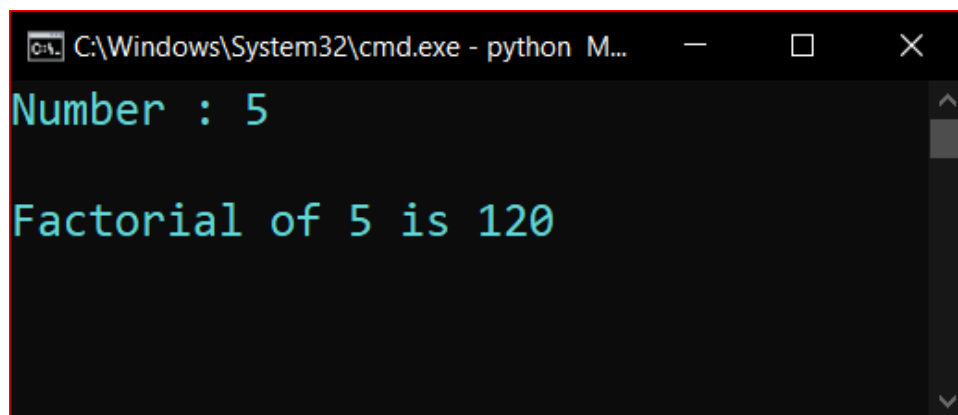
    /*Taking input*/
    printf("Number : ");
    scanf("%u", &number);

    factorial = fact(number);/*Function call*/

    printf("\nFactorial of %u is %u", number, factorial);/*Meaningful result*/

    getch();
    return 0;
}
```

Output:

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe - python M...'. The window has a black background with green text. The first line of output is 'Number : 5'. The second line of output is 'Factorial of 5 is 120'. There is a vertical scrollbar on the right side of the window.

```
C:\Windows\System32\cmd.exe - python M...
Number : 5
Factorial of 5 is 120
```

4. Write a program to check a function void check_prime(): The Task of this program is to read a number and check whether the number is prime or not and display the appropriate message. Be sure that a real number cannot be either prime or composite. What about negative number

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

/*Function*/
void check_prime(int a)
{
    for (int i = 2; i < a; i++)
    {
        if (a % i == 0)
        {
            printf("\nIt is compostite number");
            exit(0);
        }
    }
    printf("\nIt is prime number");
}

int main()
{
    /*Variable Declaration*/
    int number;
```

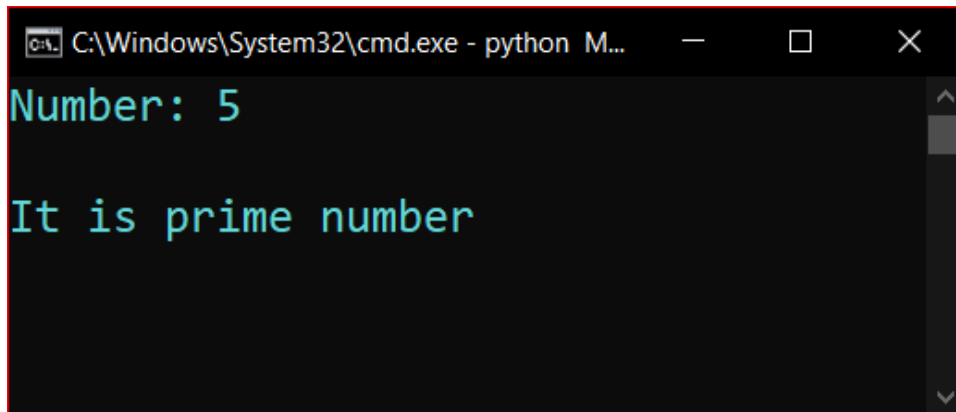
```
system("cls");/*clear the screen*/

/*Taking input*/
printf("Number: ");
scanf("%d", &number);

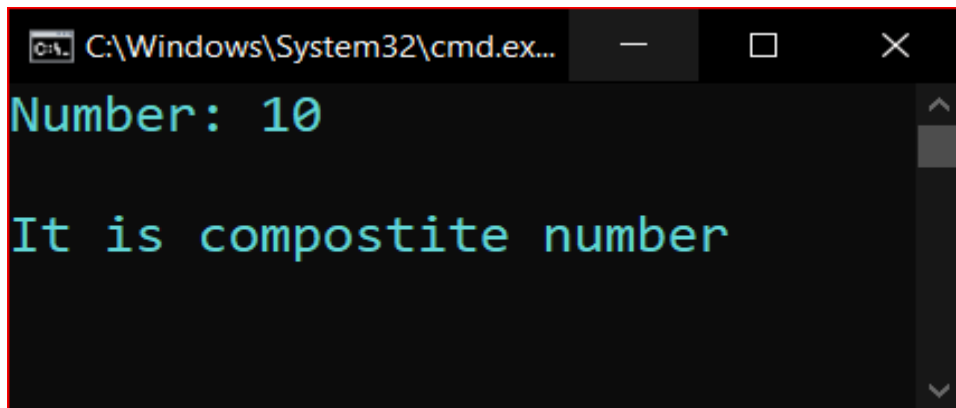
check_prime(number);/*Function call*/

getch();
return 0;
}
```

Output:



A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe - python M...'. The window has a black background with green text. It displays 'Number: 5' on the first line and 'It is prime number' on the second line. A vertical scrollbar is visible on the right side.



A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe...'. The window has a black background with green text. It displays 'Number: 10' on the first line and 'It is compostite number' on the second line. A vertical scrollbar is visible on the right side.

5.Combine Question 1 2 3 4 using switch statement. For this display a menu on the screen to prompt user whether he wants to sum two number or sumof digits of an integer or calculate factorial of an integer or to know whether number is prime or not

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
/*Function taken from Program 1,2,3,4*/
```

```
float add(int a, float b)
```

```
{
```

```
    return (float)a + b;
```

```
}
```

```
int sumOfDigits(int a)
```

```
{
```

```
    if (a < 0)
```

```
    {
```

```
        return 0;
```

```
    }
```

```
    else
```

```
    {
```

```
        while (a != 0)
```

```
        {
```

```
            int rem = a % 10;
```



```
        a = a / 10;
        return rem + sumOfDigits(a);
    }
}
}
```

```
unsigned int fact(unsigned int a)
{
    if (a == 0)
    {
        return 1;
    }
    return a * fact(a - 1);
}
```

```
void check_prime(int a)
{
    for (int i = 2; i < a; i++)
    {
        if (a % i == 0)
        {
            printf("\nIt is compostite number");
            exit(0);
        }
    }
    printf("\nIt is prime number");
}
```

*/*Function to display Option*/*

void menu()

{

printf("<-----Menu----->\n");

printf("%5s%s\n", "", "1.Sum of Numbers");

printf("%5s%s\n", "", "2.Sum of Digits");

printf("%5s%s\n", "", "3.Factorial");

printf("%5s%s\n", "", "4.Check Prime or Composite");

}

int main()

{

*/*Variable Declaration*/*

int one_number_one;

float one_number_two, one_sum;

int two_number, two_sum;

unsigned three_number, three_factorial;

int four_number;

int choice;

*/*Clear the screen*/*

system("cls");

```
menu();/*Function call*/
```

```
/*Taking input*/
```

```
printf("Choice: ");
```

```
scanf("%d", &choice);
```

```
/*Simple switch case and in every case code copy from Program 1,2,3,4*/
```

```
switch (choice)
```

```
{
```

```
case 1:
```

```
system("cls"); /*Clear the Screen*/
```

```
/*Taking input*/
```

```
printf("Number one:");
```

```
scanf("%d", &one_number_one);
```

```
printf("Number two:");
```

```
scanf("%f", &one_number_two);
```

```
/*Function call*/
```

```
one_sum = add(one_number_one, one_number_two);
```

```
/*Display the result*/
```

```
printf("\nAddition of %d and %.2f is %.2f", one_number_one, one_number_two,  
one_sum);
```

break;

case 2:

system("cls");

printf("Number: ");

scanf("%d", &two_number);

two_sum = sumOfDigits(two_number);

printf("Sum of digits in %d is %d", two_number, two_sum);

case 3:

system("cls");

printf("Number : ");

scanf("%u", &three_number);

three_factorial = fact(three_number);

printf("\nFactorial of %u is %u", three_number, three_factorial);

break;

case 4:

```
system("cls");
```

```
printf("Number: ");
```

```
scanf("%d", &four_number);
```

```
check_prime(four_number);
```

```
default:
```

```
printf("You didn't choose right option.");
```

```
break;
```

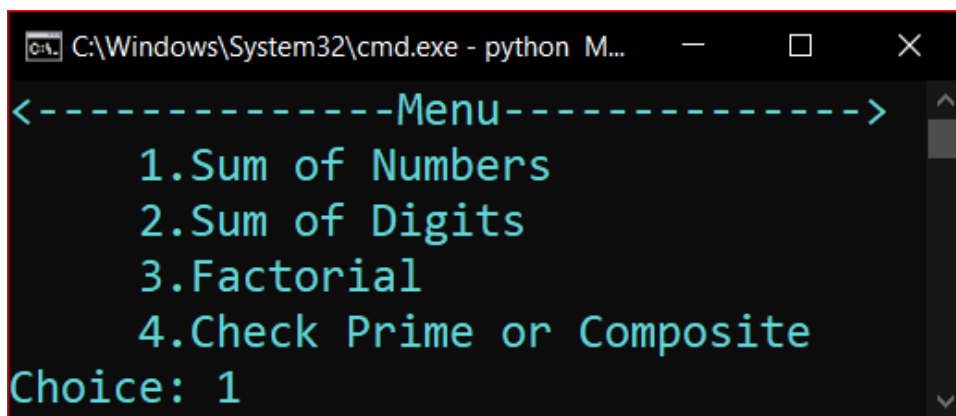
```
}
```

```
getch();
```

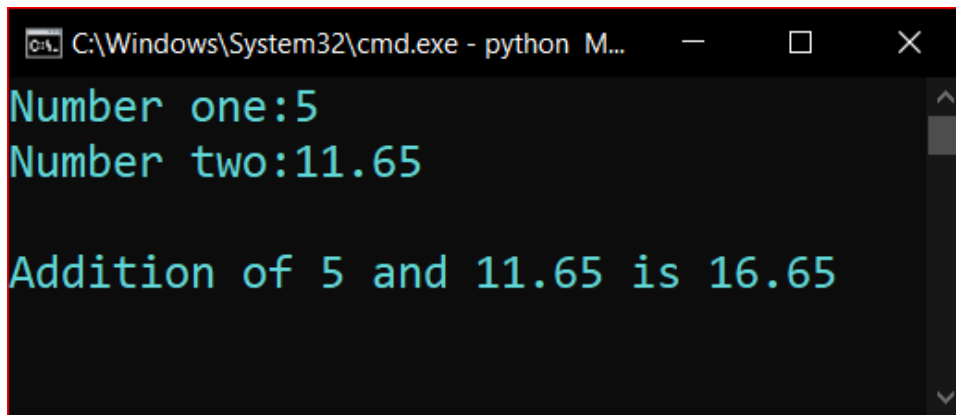
```
return 0;
```

```
}
```

Output:



```
C:\Windows\System32\cmd.exe - python M...  
<-----Menu----->  
1.Sum of Numbers  
2.Sum of Digits  
3.Factorial  
4.Check Prime or Composite  
Choice: 1
```



```
C:\Windows\System32\cmd.exe - python M...
Number one:5
Number two:11.65

Addition of 5 and 11.65 is 16.65
```

6. Write a program to read an unsigned integer in main() pass it to function. (void countsDigits(int*, int *);). This function counts the number of odd and even digits in it. Display the count from main. Use concept of passing argument by reference

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

/*Function definition*/

void countsDigits(int *, int *);

int main()
{
    /*Variable declaration*/
    int digits, even_count = 0, odd_count = 0;

    system("cls"); /*Clear the screen*/

    /*Taking input*/
    printf("Number: ");
    scanf("%d", &digits);
```

```
even_count = digits;
```

```
/*Function call*/
```

```
countsDigits(&even_count, &odd_count);
```

```
/*Meaningful result*/
```

```
printf("\nEven Digits: %d\n", even_count);
```

```
printf("Odd Digits: %d", odd_count);
```

```
getch();
```

```
return 0;
```

```
}
```

```
/*Function Body*/
```

```
void countsDigits(int *e, int *o)
```

```
{
```

```
int number = *e;
```

```
*e = 0;
```

```
int even = 0, odd = 0;
```

```
while (number != 0)
```

```
{
```

```
if (number % 2 == 0)
```

```
{
```

```
even++;
```

```
}
```

```
else if (number % 2 == 1)
```

```

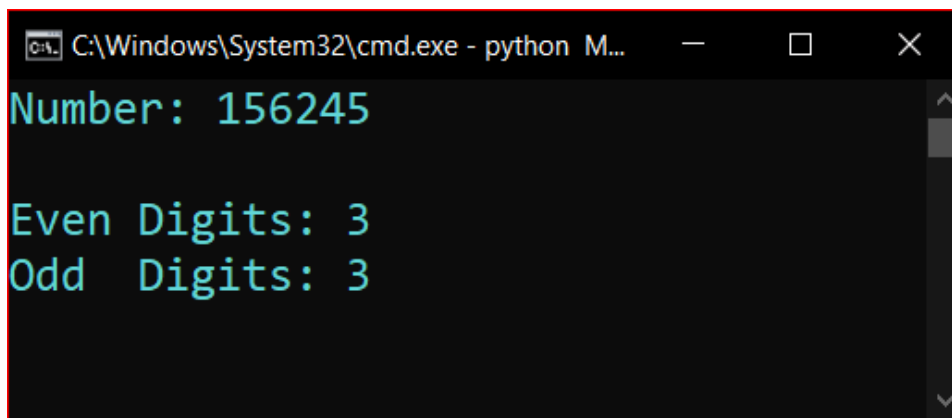
    {
        odd++;
    }

    number /= 10;
}

*e = even;
*o = odd;
}

```

Output:



```

C:\Windows\System32\cmd.exe - python M...
Number: 156245
Even Digits: 3
Odd Digits: 3

```

7. Write a program to create a function: `int findLowest(int,int ,int);` and `int findHighest(int,int,int);` The task of `findLowest()` is to find the lowest of three integers and return an integer to the calling function. Similarly, the task of `findHighest()` is to find the highest of three integers and return an integer to the calling function. Call these functions in `main()` giving appropriate arguments.

Note: Use conditional operator (`test expression? expression1: expression2`) to find highest and lowest

Source Code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

/*Function definition*/

```



```
int findLowest(int, int, int);
```

```
int findHighest(int, int, int);
```

```
int main()
```

```
{
```

```
/*Variable Declaration*/
```

```
int number_one, number_two, number_three;
```

```
system("cls");/*Clear the screen*/
```

```
/*Taking Input*/
```

```
printf("Number One: ");
```

```
scanf("%d", &number_one);
```

```
printf("Number Two: ");
```

```
scanf("%d", &number_two);
```

```
printf("Number Three: ");
```

```
scanf("%d", &number_three);
```

```
/*Function call and printing*/
```

```
printf("\nLowest    number    is    %d",    findLowest(number_one,    number_two,  
number_three));
```

```
printf("\nHighest    number    is    %d",    findHighest(number_one,    number_two,  
number_three));
```

```
getch();
```

```
return 0;
```

```
}
```

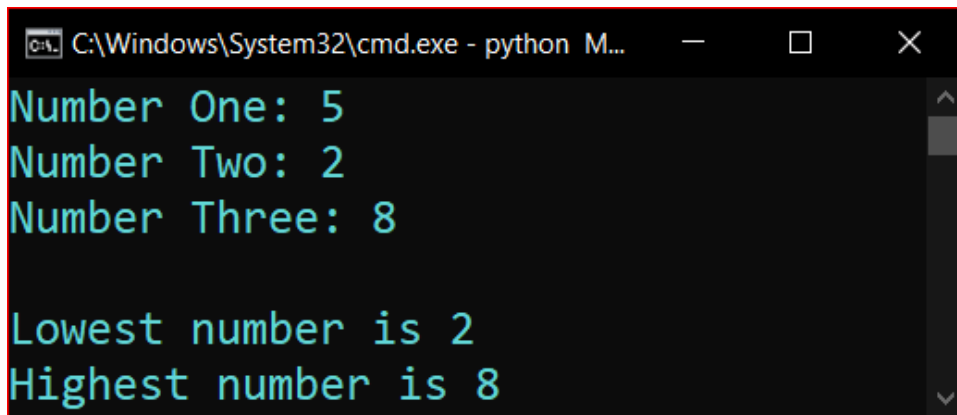
```

/*Function body*/
int findLowest(int a, int b, int c)
{
    int low;
    (a < b && a < c) ? ((low = a)) : ((b < c) ? ((low = b)) : ((low = c)));
    return low;
}

int findHighest(int a, int b, int c)
{
    int high;
    (a > b && a > c) ? ((high = a)) : ((b > c) ? ((high = b)) : ((high = c)));
    return high;
}

```

Output:



```

C:\Windows\System32\cmd.exe - python M...
Number One: 5
Number Two: 2
Number Three: 8

Lowest number is 2
Highest number is 8

```

8.1 Factorial of n by recursive function

Source Code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

```

*/*Fucntion definition and declaration*/*

int factorial(int n)

{

if (n == 0)

{

return 1;

}

else

{

*return n * factorial(n - 1);*

}

}

int main()

{

*/*Variable Declaration*/*

int number;

*system("cls");/*Clear the screen*/*

*/*Taking input from user*/*

printf("Number: ");

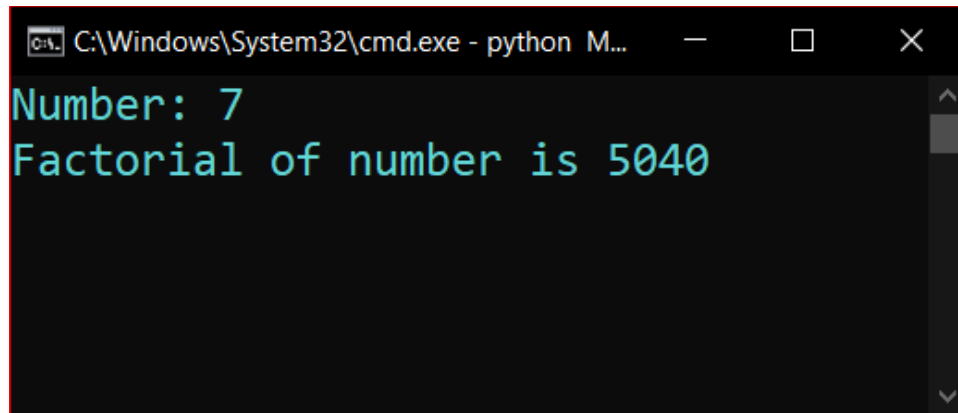
scanf("%d", &number);

*/*Printing result*/*

(number < 0) ? ((printf("Invalid number.))) : (printf("Factorial of number is %d", factorial(number)));

```
    getch();  
    return 0;  
}
```

Output:

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe - python M...'. The window has a black background with green text. It displays 'Number: 7' on the first line and 'Factorial of number is 5040' on the second line. A vertical scrollbar is visible on the right side of the window.

```
C:\Windows\System32\cmd.exe - python M...  
Number: 7  
Factorial of number is 5040
```

8.2 Compute x^n

Source Code:

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
  
/*Fucnction definition and declaration*/  
int powerOfBase(int base, int power)  
{  
    if (power == 0)  
    {  
        return 1;  
    }  
    else  
    {
```

```

        return base * powerOfBase(base, power - 1);
    }
}

int main()
{
    /*Variable declaration*/
    int x, n;

    system("cls");/*Clear the screen*/

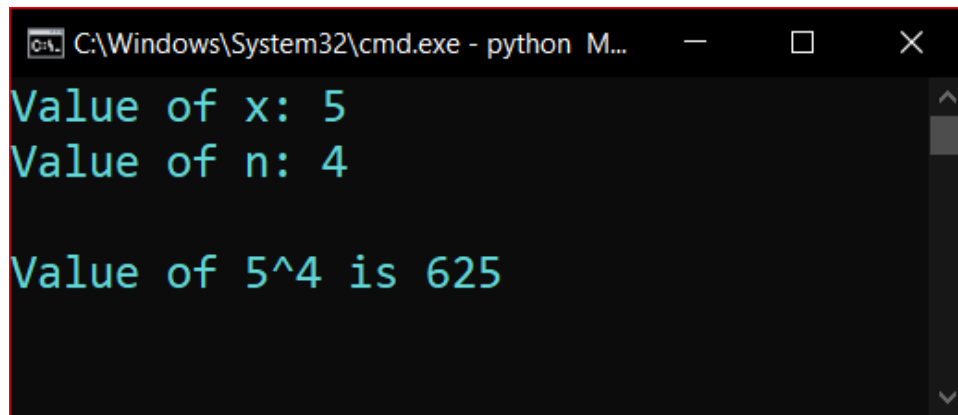
    /*Taking user input*/
    printf("Value of x: ");
    scanf("%d", &x);
    printf("Value of n: ");
    scanf("%d", &n);

    /*Meaningful Output*/
    printf("\nValue of %d^%d is %d", x, n, powerOfBase(x, n));

    getch();
    return 0;
}

```

Output:



```
C:\Windows\System32\cmd.exe - python M...  
Value of x: 5  
Value of n: 4  
  
Value of 5^4 is 625
```

8.3HCF of number

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
/*Function defination and declaration*/
```

```
int HCF(int a, int b)
```

```
{
```

```
/*Below if just swap the value of a and b.For more clarity, once look  
Swap_two_numbers.c*/
```

```
if(a < b)
```

```
{
```

```
    b = a + b - (b = a);
```

```
}
```

```
/*Termination condition of recursion*/
```

```
if(b == 0)
```

```
{  
    return a;  
}
```

```
/*Declaration of variable*/
```

```
int quotient;
```

```
int remainder;
```

```
/*Explained Above*/
```

```
quotient = a / b;
```

```
remainder = a - quotient * b;
```

```
/*Calling to it self*/
```

```
HCF(b, remainder);
```

```
}
```

```
int main()
```

```
{
```

```
/*Variable declaration*/
```

```
int a, b;
```

```
system("cls");/*Clear the screen*/
```

```
/*Taking input*/
```

```
printf("Number One: ");
```

```
scanf("%d", &a);
```

```

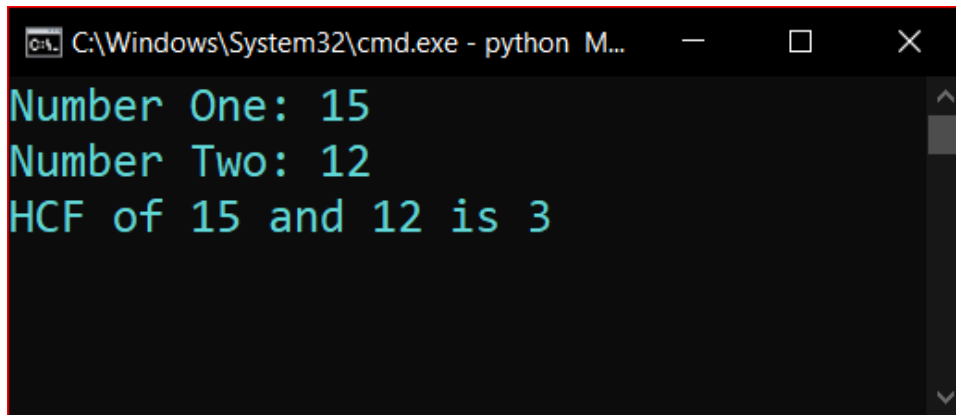
printf("Number Two: ");
scanf("%d", &b);

/*Meaningful result*/
printf("HCF of %d and %d is %d", a, b, HCF(a, b));

getch();
return 0;
}

```

Output:



```

C:\Windows\System32\cmd.exe - python M...
Number One: 15
Number Two: 12
HCF of 15 and 12 is 3

```

8.4Sum from 1 to n

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

/*Fucntion declaration*/
int sum(int n)
{
    if (n == 0)

```



```

    {
        return 0;
    }
    else
    {
        return n + sum(n - 1);
    }
}

int main()
{
    /*Variable declaration*/
    int n;

    system("cls");/*Clear the screen*/

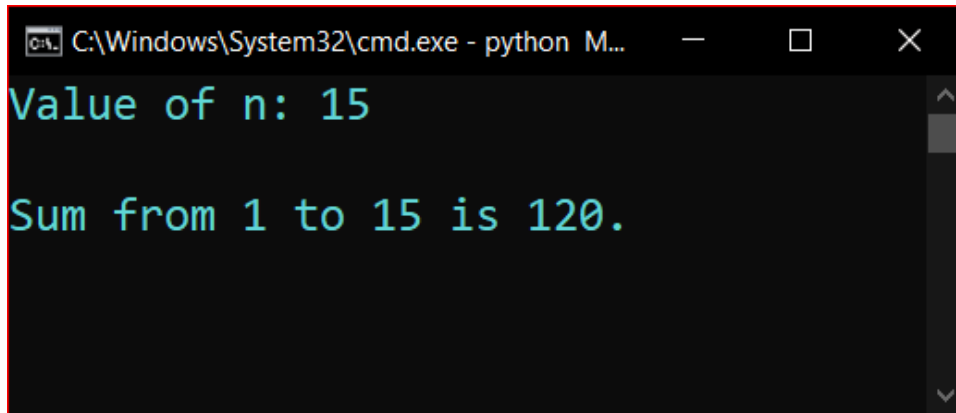
    /*Taking user input*/
    printf("Value of n: ");
    scanf("%d", &n);

    /*Meaningful output*/
    printf("\nSum from 1 to %d is %d.", n, sum(n));

    getch();
    return 0;
}

```

Output:



```
C:\Windows\System32\cmd.exe - python M...
Value of n: 15
Sum from 1 to 15 is 120.
```

9. Write a program using recursive function to compute series $1^2 - 2^2 + 3^2 - \dots - (-1)^{(n+1)} \cdot n^2$

You cannot use pow function

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

/*Function declaration*/
float series(int n)
{
    if (n == 1)
    {
        return 1;
    }
    else
    {
        return (n % 2 == 1) ? (n * n + series(n - 1)) : (-1 * n * n + series(n - 1));
    }
}
```

```
int main()
{
    /*Variable declaration*/
    int n;

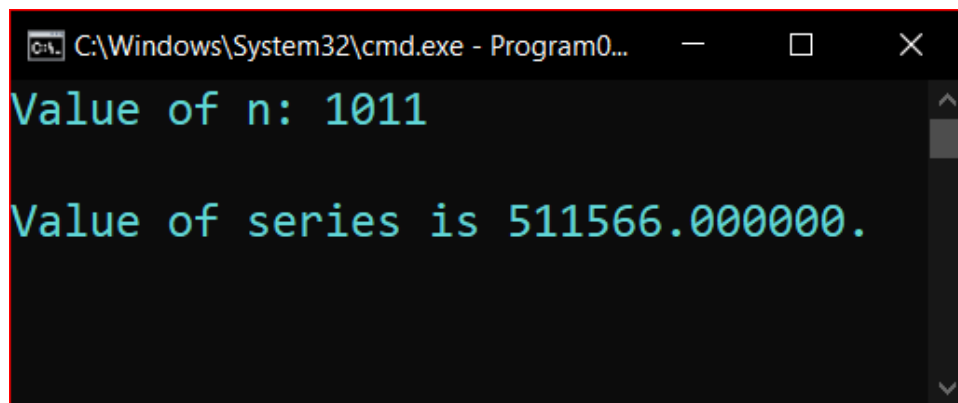
    system("cls");/*Clear the screen*/

    /*Taking user input*/
    printf("Value of n: ");
    scanf("%d", &n);

    /*Output*/
    printf("\nValue of series is %f.", series(n));

    getch();
    return 0;
}
```

Output:

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe - Program0...'. The window has a black background with green text. The first line of output is 'Value of n: 1011'. The second line of output is 'Value of series is 511566.000000.'. There are up and down arrow icons on the right side of the window.

```
C:\Windows\System32\cmd.exe - Program0...
Value of n: 1011
Value of series is 511566.000000.
```

Analysis:

Here we learn about declaring function in C language how to user it. Along with declaration of function we also came to know the working nature of function. We also came how does pass by value and pass by reference works.

Conclusion

Here we learn to work with function.

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 8

Title: **Arrays**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

C Array

Code and Output

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

- In the Windows search bar, type 'settings' to open your Windows Settings.
- Search for Edit environment variables for your account.
- Choose the Path variable and then select Edit.
- Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
- Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

What is Array?

An array is a collection of one or more values of the same type. Each value is called an element of the array. The elements of the array share the same variable name but each element has its own unique index number (also known as a subscript). An array can be of any type, For example: int, float, char etc. If an array is of type int then its elements must be of type int only

To store roll no. of 100 students, we have to declare an array of size 100 i.e roll_no[100]. Here size of the array is 100, so it is capable of storing 100 values. In C, index or subscript starts from 0, so roll_no[0] is the first element, roll_no[1] is the second element and so on. Note that the last element of the array will be at roll_no[99] not at roll_no[100] because the index starts at 0.

Arrays can be single or multidimensional. The number of subscript or index determines the dimensions of the array. An array of one dimension is known as a one-dimensional array or 1-D array, while an array of two dimensions is known as a two-dimensional array or 2-D array.

Let's start with a one-dimensional array.

One Dimensional Array

Conceptually you can think of a one-dimensional array as a row, where elements are stored one after another.

Syntax:

```
datatype array_name[size];
```

datatype: It denotes the type of the elements in the array.

array_name: Name of the array. It must be a valid identifier.

size: Number of elements an array can hold. here are some example of array declarations:

```
int num[100];float temp[20];char ch[50];
```

num is an array of type int, which can only store 100 elements of type int.

temp is an array of type float, which can only store 20 elements of type float.

ch is an array of type char, which can only store 50 elements of type char.

Note: When an array is declared it contains garbage values.

Two Dimensional Array

The syntax declaration of 2-D array is not much different from 1-D array. In 2-D array, to declare and access elements of a 2-D array we use 2 subscripts instead of 1.

Syntax: `datatype array_name[ROW][COL];`

The total number of elements in a 2-D array is ROW*COL. Let's take an example.

```
int arr[2][3];
```

This array can store $2*3=6$ elements. You can visualize this 2-D array as a matrix of 2 rows and 3 columns.

The individual elements of the above array can be accessed by using two subscript instead of one. The first subscript denotes row number and second denotes column number. As we can see in the above image both rows and columns are indexed from 0. So the first element of this array is at arr[0][0] and the last element is at arr[1][2]. Here are how you can access all the other elements:

arr[0][0] - refers to the first element

arr[0][1] - refers to the second element

arr[0][2] - refers to the third element

arr[1][0] - refers to the fourth element

arr[1][1] - refers to the fifth element

arr[1][2] - refers to the sixth element

If you try to access an element beyond valid ROW and COL , C compiler will not display any kind of error message, instead, a garbage value will be printed. It is the responsibility of the programmer to handle the bounds.

arr[1][3] - a garbage value will be printed, because the last valid index of COL is 2

arr[2][3] - a garbage value will be printed, because the last valid index of ROW and COL is 1 and 2 respectively

Just like 1-D arrays, we can only also use constants and symbolic constants to specify the size of a 2-D array.

One Dimension Array

1]Write,observe and study the following code

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```

void main()
{

    int i, num[6] = {4, 5, 3, 2, 12,8};

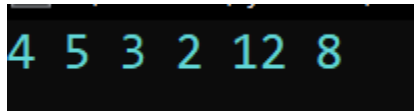
    system("cls");

    for (int i = 0; i < 6; i++)
    {
        printf("%d ", num[i]);
    }

    getch();
}

```

Output:



```

4 5 3 2 12 8

```

2].Write,observe and study the following code

Source Code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main()
{
    int i, num[6];

```

```

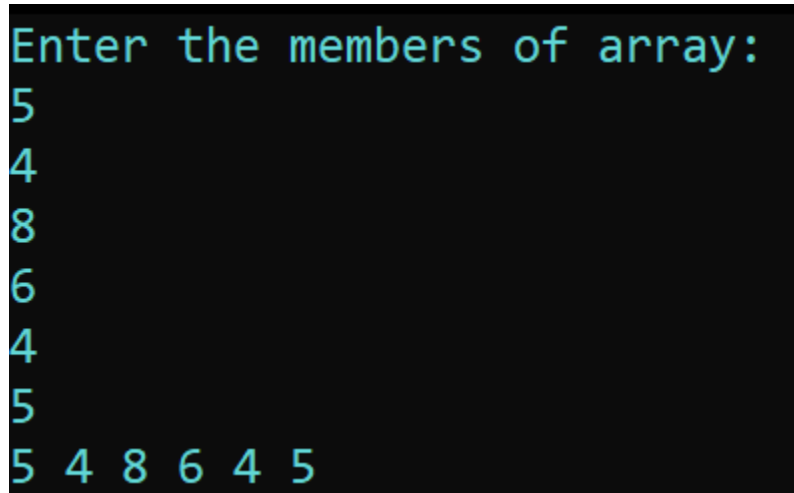
system("cls");

printf("Enter the members of array: \n");
for (int i = 0; i < 6; i++)
{
    scanf("%d", &num[i]);
}
for (int i = 0; i < 6; i++)
{
    printf("%d ", num[i]);
}

getch();
return 0;
}

```

Output:



```

Enter the members of array:
5
4
8
6
4
5
5 4 8 6 4 5

```

3]Write a program find the sum of elements of an integers array of size 5 that are divisible by 10 not by 15.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    /*Declaration of variable*/
    int i, num[5], sum = 0;

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: ");
    for (int i = 0; i < 5; i++)
    {
        scanf("%d", &num[i]);
    }

    /*Check the condition of every number and adding them if they meet condition asked by question*/
    for (int i = 0; i < 5; i++)
    {
        (num[i] % 10 == 0 && num[i] % 15 != 0) ? (sum = num[i] + sum) : (sum = sum);
    }

    /*Printing the result*/
    printf("The sum of number present in the array divisible by 10 not by 15 is %d",sum);

    getch();
    return 0;
}
```

Output:

```
Enter the members of array: 20
40
30
45
60
The sum of number present in the array divisible by 10 not by 15 is 60.
```

4]Write a program to add the elements at the corresponding position of two array of size n. Read value of n from user

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#include "../Array.c" /*External file which contain related to one d array*/

int main()
{
    /*Declaration of needed variable*/
    int number1[100], number2[100], number[100], size;

    system("cls");

    /*Asking the size of array*/
    printf("Size of the array: ");
    scanf("%d", &size);

    /*Taking input of array one and array two*/
    printf("\nArray one:\n");
    inputOneDArray(number1, size);

    printf("\nArray Two:\n");
    inputOneDArray(number2, size);
```

```
/*Adding both element of array*/
for (int i = 0; i < size; i++)
{
    number[i] = number1[i] + number2[i];
}

/*Displaying result*/
printf("\nSum of Two Array:\n");
displayOneDArray(number, size);

getch();
return 0;
}
```

Output:

```
Size of the array: 4

Array one:
Value of 1 element is 4
Value of 2 element is 5
Value of 3 element is 6
Value of 4 element is 7

Array Two:
Value of 1 element is 1
Value of 2 element is 5
Value of 3 element is 8
Value of 4 element is 7

Sum of Two Array:
5 10 14 14
```

5]Write a program to find the highest or lowest elements of an array of size 5

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    /*Declaration of variable*/
    int i, num[5], lowest, highest;

    system("cls");
```



```

/*Taking input from user*/

printf("Enter the members of array: \n");
for (int i = 0; i < 5; i++)
{
    scanf("%d", &num[i]);
}

/*Initialization of lowest and highest*/

lowest = num[0];
highest = num[0];

/*Iterate over all elements of array and find lowest and highest number from array and store to a variable*/

for (int i = 1; i < 6; i++)
{
    (lowest < num[i]) ? (lowest = lowest) : (lowest = num[i]);
    (highest > num[i]) ? (highest = highest) : (highest = num[i]);
}

/*Printing result*/

printf("\nHighest Number is %d\n", highest);
printf("Lowest Number is %d", lowest);

getch();

return 0;
}

```

Output:

```
Enter the members of array:
45
1245
14
458
45

Highest Number is 1245
Lowest Number is 5
```

6]Write a program to read the element of array in main() pass it to fucntion to sort the array in ascending/descending order. Display the sorted array from main().

Source Code

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
void sortArray(int num[], char sort[5])
```

```
{
```

```
    "Sort the array in ascending or descending order based on second argument passed by  
    user. Here, we use bubble sort algorithm.";
```

```
    int temp;
```

```
    for (int i = 0; i < 5; i++)
```

```
{
```

```
for (int j = 0; j < 5; j++)
{
    if (strcmp(sort, "Ascend") == 0)
    {
        if (num[i] > num[j])
        {
        }
        else
        {
            temp = num[i];
            num[i] = num[j];
            num[j] = temp;
        }
    }
    else if (strcmp(sort, "Descend") == 0)
    {
        if (num[i] < num[j])
        {
        }
        else
        {
            temp = num[i];
            num[i] = num[j];
            num[j] = temp;
        }
    }
}
```

```

    }
}

int main()
{
    /*Declaration of variable*/
    int i, num[5], lowest, highest;

    system("cls");

    /*Taking input from the user*/
    printf("Enter the members of array: ");
    for (int i = 0; i < 5; i++)
    {
        scanf("%d ", &num[i]);
    }

    sortArray(num, "Ascend"); /*Function call(pass by refrence) to sort array ascending order*/

    /*Printing the sorted array*/
    printf("\nSorted in Ascending Order: ");
    for (int i = 0; i < 5; i++)
    {
        printf("%d ", num[i]);
    }
}

```

```
    sortArray(num, "Descend"); /*Function call(pass by refrence) to sort array
descending order*/
```

```
/*Printing the sorted array*/
```

```
printf("\n\nSorted in Descending Order: ");
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

```
    printf("%d ", num[i]);
```

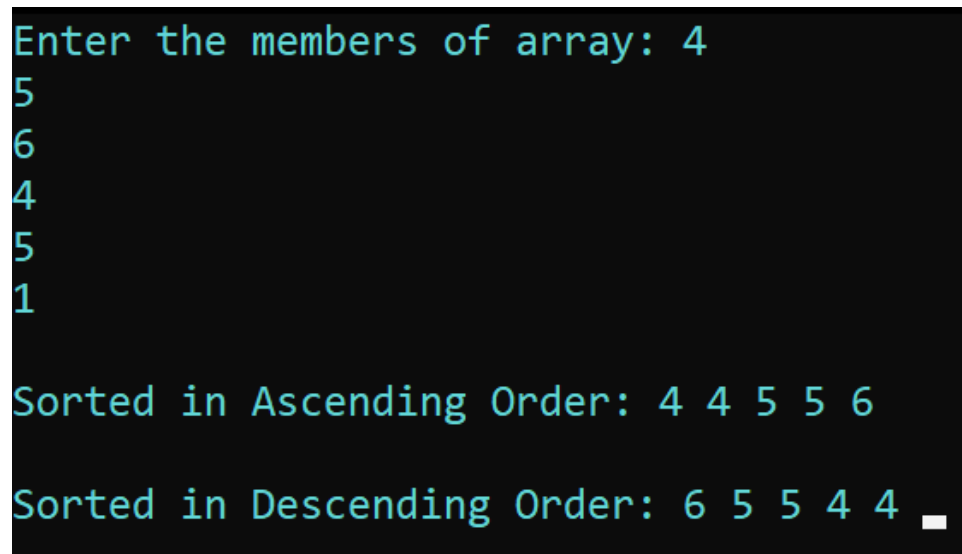
```
}
```

```
getch();
```

```
return 0;
```

```
}
```

Output:



```
Enter the members of array: 4
5
6
4
5
1

Sorted in Ascending Order: 4 4 5 5 6

Sorted in Descending Order: 6 5 5 4 4
```

7]Write a program to raise the power of each member by 3

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int main()
```

```
{
```

```
/*Declaration of variable*/
```

```
int i, num[5];
```

```
system("cls");
```

```
/*Taking input from user*/
```

```
printf("Enter the members of array: \n");
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

```
scanf("%d", &num[i]);
```

```
}
```

```
/*Raising the value of each element to power of 3 of respective element*/
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

```
num[i] = num[i] * num[i] * num[i];
```

```
}
```

```
/*Printing the new array(raised by 3)*/
```

```
printf("\nNew Array: ");
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

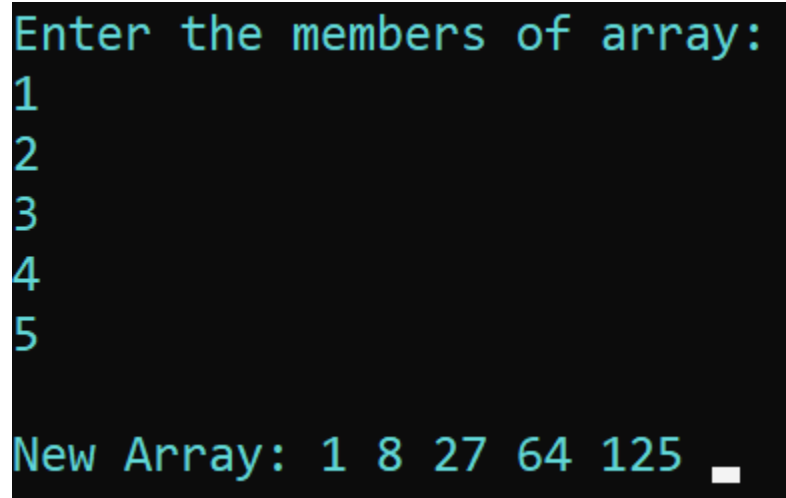
```
printf("%d ", num[i]);
```

```
}
```

```
getch();
```

```
    return 0;
}
```

Output:



```
Enter the members of array:
1
2
3
4
5

New Array: 1 8 27 64 125
```

8]Write a program to read an unsigned integer array from main and pass it to a function that count armstrong members and return the count to main()*/

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int armstrongArray(int num[])
{
    "Takes an array count the occurance of armstrong number and return it";
    int count = 0, armstrong = 0, n, rem;

    for (int i = 0; i < 5; i++)
    {
        n = num[i];
        if (n == 0)
        {
            count++;
        }
    }
}
```

```

    }
    else
    {

        while (n != 0)
        {
            rem = n % 10;

            armstrong = n * n * n + armstrong;

            if (num[i] == armstrong)
            {
                count++;
            }

            n = n / 10;
        }

        armstrong = 0;
    }
}

return count;
}

int main()
{
    /*Declaration of variable*/
    unsigned int num[5];

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: ");

```



```

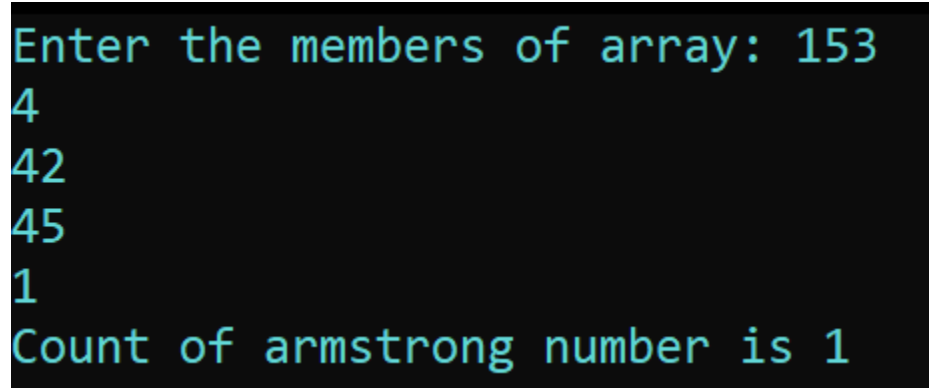
    for (int i = 0; i < 5; i++)
    {
        scanf("%d", &num[i]);
    }

    /*Print the value of count of armstrong numbers in array afer function call*/
    printf("Count of armstrong number is %d", armstrongArray(num));

    getch();
    return 0;
}

```

Output:



```

Enter the members of array: 153
4
42
45
1
Count of armstrong number is 1

```

Program Array.c

```
#include <math.h>
```

```
void inputOneDArray(int array[], int array_length)
```

```
{
```

```
    "Takes two argument array and length of array. Ask user to input the value in respective array
    element";
```

```
    for (int i = 0; i < array_length; i++)
```

```
{
```

```
        printf("Value of %d element is ", i + 1);
```

```

        scanf("%d", &array[i]);
    }
}

void sortOneDArrayAscend(int array[], int array_length)
{
    "Sorted the array in ascending order";

    for (int i = 0; i < array_length; i++)
    {
        for (int j = 0; j < array_length; j++)
        {
            if (array[i] < array[j])
            {
                array[i] = array[i] + array[j] - (array[j] = array[i]);
            }
        }
    }
}

```

```

void sortOneDArrayDescend(int array[], int array_length)
{
    "Sorted the array in ascending order";

    for (int i = 0; i < array_length; i++)
    {
        for (int j = 0; j < array_length; j++)
        {
            if (array[i] > array[j])
            {

```

```

        array[i] = array[i] + array[j] - (array[j] = array[i]);
    }
}
}
}

```

```
float medianOneDArray(int array[], int array_length)
```

```

{
    "Calculate median and return value of median";

    float position_of_array = (array_length + 1) / 2.0;

    if (position_of_array - (int)position_of_array == 0.0)
    {
        return array[(int)position_of_array - 1];
    }
    else
    {
        return array[(int)position_of_array] + array[(int)position_of_array - 1] / 2.0;
    }
}

```

```
int smallOneDArray(int array[], int array_length)
```

```

{
    "Finds the smallest elements in array";

    int small = array[0];

    for (int i = 1; i < array_length; i++)
    {

```

```
        if (small > array[i])
        {
            small = array[i];
        }
    }

    printf("small = %d\n", small);

    return small;
}

int largeOneDArray(int array[], int array_length)
{
    "Finds the largest element of array";

    int large = array[0];

    for (int i = 1; i < array_length; i++)
    {
        if (large < array[i])
        {
            large = array[i];
        }
    }

    printf("large = %d\n", large);

    return large;
}
```

```
int sumOneDArray(int array[], int array_length)
```

```
{
```

```
    "Calculates the sum of elements of array";
```

```
    int sum = 0;
```

```
    for (int i = 0; i < array_length; i++)
```

```
    {
```

```
        sum = sum + array[i];
```

```
    }
```

```
    return sum;
```

```
}
```

```
float meanOneDArray(int array[], int array_length)
```

```
{
```

```
    "Calculate the mean of elements of integer array";
```

```
    float mean = 0.0;
```

```
    mean = sumOneDArray(array, array_length) / (float)array_length;
```

```
    return mean;
```

```
}
```

```
float sdOneDArray(int array[], int array_length)
```

```
{
```

```
    "Calculates the standard deviation of elements of integer array";
```

```
    float sd, diverse;
```

```

float mean = meanOneDArray(array, array_length);

for (int i = 0; i < array_length; i++)
{
    diverse = (mean - array[i]) * (mean - array[i]) + diverse;
}

sd = pow(diverse / array_length, 0.5);

return sd;
}

float varianceOneDArray(int array[], int array_length)
{
    "Calculate the variance of elements of integer array";

    float variance = pow(sdOneDArray(array, array_length), 2);

    return variance;
}

void displayOneDArray(int array[], int array_length)
{
    "Display the array elements";

    for (int i = 0; i < array_length; i++)
    {
        printf("%d ", array[i]);
    }
}

```

```
}
```

9.1]Program to compute median of number in array*/

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define ARRAY_LENGTH 9 /*Defining a constant used in variable*/

#include "../Array.c" /*External files that help to do operation in one d array*/

int main()
{
    /*Array declartion*/
    int median[ARRAY_LENGTH];

    system("cls");

    /*Function call to take value of array from user*/
    inputOneDArray(median, ARRAY_LENGTH);

    /*Function call to sort the array in ascending order*/
    sortOneDArrayAscend(median, ARRAY_LENGTH);

    /*Function call to find the median of number present in the array and print the value of median*/
    printf("The median of array provided is %.2f.", medianOneDArray(median, ARRAY_LENGTH));

    getch();
    return 0;
```

```
}
```

Output:

```
Value of 1 element is 4
Value of 2 element is 8
Value of 3 element is 7
Value of 4 element is 2
Value of 5 element is 6
Value of 6 element is 9
Value of 7 element is 4
Value of 8 element is 3
Value of 9 element is 8
The median of array provided is 6.00.
```

9.2]Program to compute range of one d array

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define ARRAY_LENGTH 8 /*Defining constant*/

#include "../Array.c" /*External files that helps to do operation in one d array*/

int main()
{
    /*Variable declaration*/
    int input[ARRAY_LENGTH], small, large, range;

    /*Taking input from user*/
    inputOneDArray(input, ARRAY_LENGTH);
```



```

/*Function call to find smallest value and storing value in small*/
small = smallOneDArray(input, ARRAY_LENGTH);

/*Function call to find largest value and storing value in large*/
large = largeOneDArray(input, ARRAY_LENGTH);

/*Calculating range*/
range = (large - small);

/*Printing result*/
printf("Range of given data is %d", range);

getch();
return 0;
}

```

Output:

```

Value of 1 element is 1
Value of 2 element is 5
Value of 3 element is 8
Value of 4 element is 7
Value of 5 element is 6
Value of 6 element is 9
Value of 7 element is 15
Value of 8 element is 14
small = 1
large = 15
Range of given data is 14

```

9.3]Program to compute standard deviation

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>


#define ARRAY_LENGTH 5 /*Defining a constant*/


#include "../Array.c"/*External file which have function to do operations in one d array*/


int main()
{
    /*Variable declaration*/
    int input[ARRAY_LENGTH];
    float sd;


    /*Calling function to take input*/
    inputOneDArray(input, ARRAY_LENGTH);


    system("cls");


    /*Calling function to calculate Standard Deviation*/
    sd = sdOneDArray(input, ARRAY_LENGTH);


    /*Printing the result*/
    printf("Standard Deviation is %f\n", sd);


    getch();
    return 0;
}
```

Output:

```
Value of 1 element is 10
Value of 2 element is 48
Value of 3 element is 95
Value of 4 element is 15
Value of 5 element is 42
```

```
Standard Deviation is 30.324907
```

9.4]Program to compute variance

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define ARRAY_LENGTH 5 /*Defining constant*/

#include "../Array.c" /*External files that contain function to do operation in one d array*/

int main()
{
    /*Variable Declaration*/
    float variance;
    int input[10];

    /*Calling function to take input*/
    inputOneDArray(input, ARRAY_LENGTH);

    system("cls");
```

```

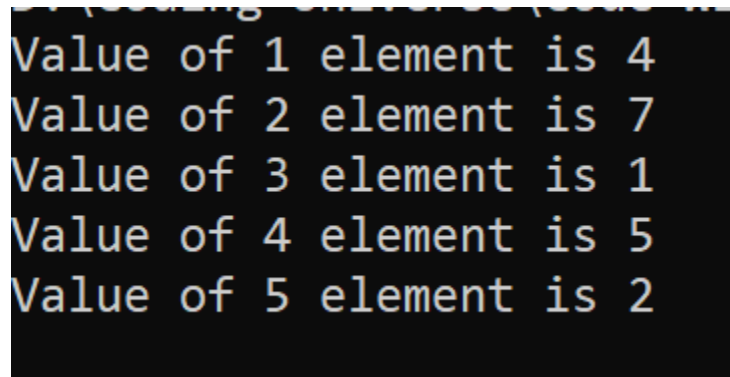
    /*Calling function to calculated variance*/
    variance = varianceOneDArray(input, ARRAY_LENGTH);

    /*Printing result*/
    printf("Variance is %.2f", variance);

    getch();
    return 0;
}

```

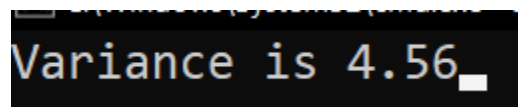
Output:



```

Value of 1 element is 4
Value of 2 element is 7
Value of 3 element is 1
Value of 4 element is 5
Value of 5 element is 2

```



```

Variance is 4.56

```

Two Dimensional Array

1]Type, run and observe the work flow

Source:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

```

```

void main()
{

```

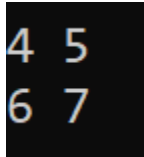
```
int i, j, num[2][2] = {{4, 5}, {6, 7}};
```

```
for (int i = 0; i < 2; i++)  
{  
    for (int j = 0; j < 2; j++)  
    {  
        printf("%d ", num[i][j]);  
    }  
    printf("\n");  
}
```

```
getch();
```

```
}
```

Output:



```
4 5  
6 7
```

2]Type, run and observe the work flow

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
void main()
```

```
{
```

```
    int i, j, num[3][3];
```

```
    for (int i = 0; i < 3; i++)
```

```
    {
```

```
        for (int j = 0; j < 3; j++)
```

```
    {  
        scanf("%d", &num[i][j]);  
    }  
}  
  
for (int i = 0; i < 3; i++)  
{  
    for (int j = 0; j < 3; j++)  
    {  
        printf("%d ", num[i][j]);  
    }  
    printf("\n");  
}  
getch();  
}
```

Output:

```
1
2
3

4
5
6
7
8
9
1 2 3
4 5 6
7 8 9
```

3]Write a program find the sum of elements of an integers array of size 5 that are divisible by 10 not by 15

Source Code:

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main()

{

    /*Variable declaration*/

    int i, num[3][3], sum = 0;


    system("cls");


    /*Taking input from user*/
```

```

printf("Enter the members of array:\n");

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        scanf("%d", &num[i][j]);
    }
}

/*Iterate over every element and check condition is match or not. if match then it keep on adding those
value*/

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        (num[i][j] % 7 == 0 && num[i][j] % 5 != 0) ? (sum = num[i][j] + sum) : (sum = sum);
    }
}

/*Prints the result*/

printf("The sum of number present in the array divisible by 7 not by 5 is %d", sum);

getch();

return 0;
}

```

Output:


```
Enter the members of array:
14
35
28
26
15
45
70
7
26
The sum of number present in the array divisible by 7 not by 5 is 49.
```

4]Write a program to find the highest or lowest elements of an array of size 3X3*/

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    /*Variable Declaration*/
    int i, num[3][3], lowest, highest;

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: \n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            scanf("%d", &num[i][j]);
        }
    }
}
```

```

/*Initialization of lowest and highest*/

lowest = num[0][0];
highest = num[0][0];

/*Iterate over every element and find the highest and lowest storing into a variable*/
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        (lowest < num[i][j]) ? (lowest = lowest) : (lowest = num[i][j]);
        (highest > num[i][j]) ? (highest = highest) : (highest = num[i][j]);
    }
}

/*Prints the result*/

printf("\nHighest Number is %d", highest);
printf("\n\nLowest Number is %d", lowest);

getch();

return 0;

}

```

Output:

Enter the members of array:

4

58

74

54

15

24

45

1

4

Highest Number is 74

Lowest Number is 1

6]Write a program to read the members of 3X3 array in main(). Pass the array to function that finds the sum of diagonal elements and returns to main(). Display the returned values

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int diagonalSum(int num[3][3])
```

```
{
```

```
    "Calculates the sum of elements of diagonal elements";
```

```
    int sum = 0;
```

```
    /*Iterate over every element and sum all the diagonal element i.e a(i,j) where i = j*/
```

```
    for (int i = 0; i < 3; i++)
```

```

{
    for (int j = 0; j < 3; j++)
    {
        (i == j) ? (sum = sum + num[i][j]) : (sum = sum);
    }
}

return sum;
}

int main()
{
    /*Variable declaration*/
    int i, num[3][3], lowest, highest;

    system("cls");

    /*Takes input from user*/
    printf("Enter the members of array: \n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            scanf("%d", &num[i][j]);
        }
    }

    /*Call the function and prints its result*/
    printf("Sum of diagonol elements is %d", diagonolSum(num));
}

```

Output:

```
Enter the members of array:
1
2
3
4
5
6
7
8
9
Sum of diagonal elements is 15
```

7]Write a program to raise the power of elements by 5

Source Code:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int powerOfNumber(int base, int power)
```

```
{
```

```
    "Raise the n power of number";
```

```
    int value = 1;
```

```
    for (int i = 1; i <= power; i++)
```

```
    {
```

```
        value = base * value;
```

```
    }
```

```
    return value;
```

```
}
```

```
int main()
{
    /*Variable declaration*/
    int i, num[3][3];

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: \n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            scanf("%d", &num[i][j]);
        }
    }

    /*Raising power with function calll*/
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            num[i][j] = powerOfNumber(num[i][j], 5);
        }
    }

    /*Printing the new array formed*/
    for (int i = 0; i < 3; i++)
    {
```

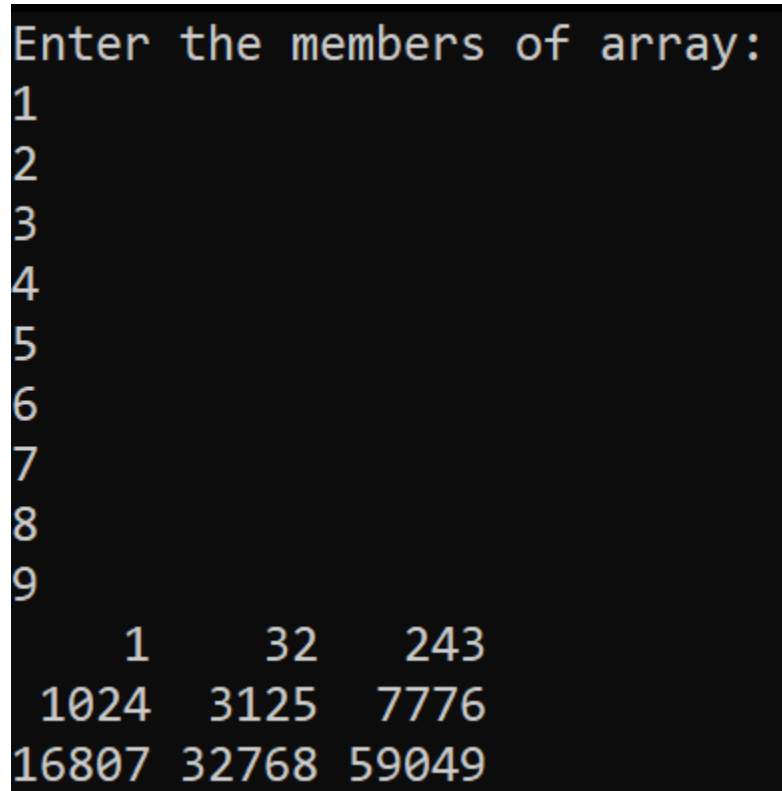
```

    for (int j = 0; j < 3; j++)
    {
        printf("%5d ", num[i][j]);
    }
    printf("\n");
}

getch();
return 0;
}

```

Output:



```

Enter the members of array:
1
2
3
4
5
6
7
8
9

      1      32      243
1024  3125  7776
16807 32768 59049

```

8] Write program to generate a matrix of size 4 * 4 whose elements are given by the expression $a(i,j) = 4^{(-1(i+j))}$

Source Code

```
#include <stdio.h>
```

```
#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    /*Declaration of variable*/
    float num[4][4];

    system("cls");

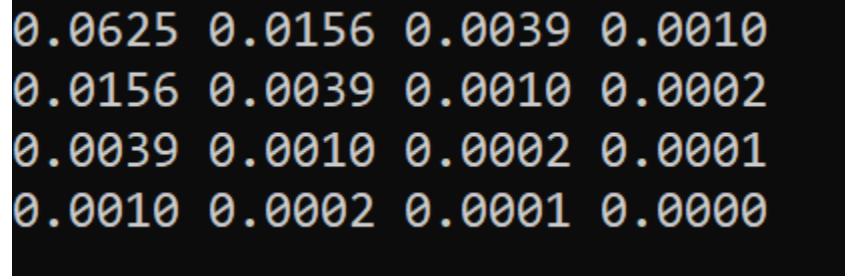
    /*Creating array from the condition given by the user*/
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            num[i][j] = pow(4, -1 * (i + j + 2));
        }
    }

    /*Printing the formed array*/
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            printf("%.4f ", num[i][j]);
        }
        printf("\n");
    }
}
```



```
    getch();  
    return 0;  
}
```

Output:



0.0625	0.0156	0.0039	0.0010
0.0156	0.0039	0.0010	0.0002
0.0039	0.0010	0.0002	0.0001
0.0010	0.0002	0.0001	0.0000

9]Write a program to find the sum of individual rows of two dimensional array and assign them to a one dimensional array and display the content of the two dimensional array

Source Code:

```
#include <stdio.h>  
  
#include <conio.h>  
  
#include <stdlib.h>  
  
  
int main()  
{  
    /*Variable declaration*/  
    int sum, numSum[10], num[10][10];  
  
    system("cls");  
  
    /*Create a array without own logic*/  
    printf("Enter the members of array: ");  
    for (int i = 0; i < 10; i++)  
    {  
        for (int j = 0; j < 10; j++)  
        {
```

```

        num[i][j] = (i)*10 + (j);
    }
}

/*Calculating the sum of element of row and storing it in a one d array*/
for (int i = 0; i < 10; i++)
{
    sum = 0;
    for (int j = 0; j < 10; j++)
    {
        sum = sum + num[i][j];
    }
    numSum[i] = sum;
}

/*Printing the two d array*/
printf("Two D Array: \n");
for (int i = 0; i < 10; i++)
{
    for (int j = 0; j < 10; j++)
    {
        printf("%3d ", num[i][j]);
    }
    printf("\n");
}

/*Printing the one d array which contain sum of elements of row*/
printf("\nSum Array: \n");
for (int i = 0; i < 10; i++)
{

```

```

        printf("%d\n", numSum[i]);
    }

    getch();

    return 0;
}

```

Output:

```

Enter the members of array: Two D Array:
 0  1  2  3  4  5  6  7  8  9
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49
50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69
70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89
90 91 92 93 94 95 96 97 98 99

Sum Array:
45
145
245
345
445
545
645
745
845
945

```

10]WAP to read matrix A and matrix B of size mXn using a function called read_matrix. Pass it to a function named process_matrix. This function should perform the multiplication of the matrices and display the result using display_matrix. You should give appropriate message to the user if multiplication of two matrix is not possible.

Source Code:

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

```

```
void readMatrix(int array[][100], int cols, int rows)
```

```
{  
    "Reads matrix value from user...";  
  
    for (int i = 0; i < rows; i++)  
    {  
        for (int j = 0; j < cols; j++)  
        {  
            printf("Value of element in %d row and %d column: ", i + 1, j + 1);  
            scanf("%d", &array[i][j]);  
        }  
    }  
}  
  
    system("cls");  
}
```

```
void processMatrix(int matrix_one[][100], int matrix_two[][100], int matrix_result[][100], int rows, int  
common, int cols)
```

```
{  
    "Multiply two matrices...";  
  
    for (int i = 0; i < rows; i++)  
    {  
        for (int j = 0; j < cols; j++)  
        {  
            matrix_result[i][j] = 0;  
            for (int k = 0; k < common; k++)  
            {  
                matrix_result[i][j] += matrix_one[i][k] * matrix_two[k][j];  
            }  
        }  
    }  
}
```

```
    }  
    }  
}  
}
```

```
void displayMatrix(int matrix [][100], int rows, int cols)
```

```
{  
    "Display the matrices...";  
  
    for (int i = 0; i < rows; i++)  
    {  
        for (int j = 0; j < cols; j++)  
        {  
            printf("%d ", matrix[i][j]);  
        }  
        printf("\n");  
    }  
}
```

```
int main ()
```

```
{  
    int matrix_one[100][100], matrix_two[100][100], matrix_multi[100][100];  
    int cols_one, rows_one, cols_two, rows_two;
```

```
system("cls");
```

```
/*Taking value of Matrix One*/
```

```
printf("Reading Matrix One: \n");
```

```
printf("Rows: ");
```

```
scanf("%d", &rows_one);
```

```

printf("Cols: ");
scanf("%d", &cols_one);
readMatrix(matrix_one, rows_one, cols_one);

/*Taking value of Matrix Two*/
printf("Reading Matrix Two: \n");
printf("Rows: ");
scanf("%d", &rows_two);
printf("Cols: ");
scanf("%d", &cols_two);
readMatrix(matrix_two, rows_two, cols_two);

if (cols_one == rows_two)
{
    processMatrix(matrix_one, matrix_two, matrix_multi, rows_one, cols_one, cols_two);
    displayMatrix(matrix_multi, rows_one, cols_two);
}
else if (cols_two == rows_one)
{
    processMatrix(matrix_two, matrix_one, matrix_multi, rows_two, cols_two, cols_one);
    displayMatrix(matrix_multi, rows_two, cols_one);
}
else
{
    printf("\nIt cannot be multiplied.");
}

Getch ();
return 0;
}

```

Output:

```
Reading Matrix One:
Rows: 2
Cols: 2
Value of element in 1 row and 1 column: 1
Value of element in 1 row and 2 column: 8
Value of element in 2 row and 1 column: 6
Value of element in 2 row and 2 column: 8_
```

```
Reading Matrix Two:
Rows: 2
Cols: 2
Value of element in 1 row and 1 column: 1
Value of element in 1 row and 2 column: 7
Value of element in 2 row and 1 column: 8
Value of element in 2 row and 2 column: 9
```

```
65 79
70 114
```

Analysis

From this lab we have learn how array is use in C programing language how it is declared initialized, how we can access it, modification and many more thng . We learn also learn what kind of operatio can be done the array. We also use 2 d array like matric and perform many calculation of matric through program.

Conclusion

From this lab we became familiar with array in C.

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 9

Title: **Pointer**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and
Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

C Pointer

Code and Output

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

- In the Windows search bar, type 'settings' to open your Windows Settings.
- Search for Edit environment variables for your account.
- Choose the Path variable and then select Edit.
- Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
- Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

POINTERS AND DYNAMIC MEMORY ALLOCATION

Pointer:

Pointer is a variable that represents the location of a data item rather than the value. A pointer variable of particular type can hold the address of its own type only.

Why is pointer useful?

- Pointers are useful as they are more efficient in handling arrays and data tables.
- It can be used to return multiple values from a function via function arguments.
- It allows passing a function as argument to other functions.
- The use of pointer arrays to character strings results in saving of data storage space in memory.
- It provides an efficient way for manipulating dynamic data structures, linked lists, queues, stacks and trees.

Pointer variable declaration:

The general syntax of pointer declaration is,

```
datatype *pointer_name;
```

Copy

The data type of the pointer and the variable to which the pointer variable is pointing must be the same.

Pointer Initialization:

Pointer Initialization is the process of assigning address of a variable to a pointer variable. It contains the address of a variable of the same data type. In C language address operator & is used to determine the address of a variable. The & (immediately preceding a variable name) returns the address of the variable associated with it.

```
int a = 10;
```

```
int *ptr;    //pointer declaration
```

```
ptr = &a;    //pointer initialization
```

Pointer expression:

Pointers are valid operands in arithmetic expressions, assignment expressions and comparison expressions. However, not all the operators normally used in these expressions are valid with pointer variables.

Arithmetic operations on pointer variables:

Declaring variables as :

```
int a,b,*p,*q;
```

- A pointer variable can be assigned the address of an ordinary variable(eg. p=&a)
- A pointer variable can be assigned the content of another pointer variable provided both pointer point to objects of same data type.(eg. p=q)
- An integer quantity can be added to or subtracted from a pointer variable. (eg. p+5, q-1, q++, --p)
- A pointer variable can be assigned a null value. (eg. p=NULL)
- One pointer variable can be subtracted from another provided both pointer point to elements of same array.

- Two pointer variables can be compared provided both pointer point to element of the same data type.

Following operations are not allowed on pointer variables:

1. Pointer variables cannot be multiplied or divided by a constant.
2. Two pointer variables cannot be added.

Dynamic memory allocation:

C dynamic memory allocation refers to performing manual memory management for dynamic memory allocation in the C programming language via a group of functions in the C standard library, namely malloc, realloc, calloc and free.

Malloc ():

The malloc() function takes a single parameter, which is the size of the requested memory area in bytes. It returns a pointer to the allocated memory. If the allocation fails, it returns NULL.

The prototype for the standard library function is like this:

```
void *malloc(size_t size);
```

Calloc():

The calloc() function does basically the same job as malloc(), except that it takes two parameters – the number of array elements and the size of each element – instead of a single parameter (which is the product of these two values). The allocated memory is also initialized to zeros. Here is the prototype:

```
void *calloc(size_t nelements, size_t elementSize);
```

Realloc():

The `realloc()` function resizes a memory allocation previously made by `malloc()`. It takes as parameters a pointer to the memory area and the new size that is required. If the size is reduced, data may be lost. If the size is increased and the function is unable to extend the existing allocation, it will automatically allocate a new memory area and copy data across. In any case, it returns a pointer to the allocated memory. Here is the prototype:

```
void *realloc(void *pointer, size_t size);
```

Free():

The `free()` function takes the pointer returned by `malloc()` and de-allocates the memory. No indication of success or failure is returned. The function prototype is like this:

```
void free(void *pointer);
```

1.1 Run the following program and see the output

Source Code

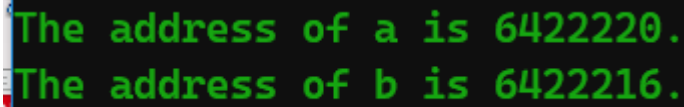
```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    int a, b;

    system("cls");

    printf("The address of a is %u.\n", &a);
    printf("The address of b is %u.", &b);
    getch();
    return 0;
}
```

Output



The address of a is 6422220.
The address of b is 6422216.

1.2 Run the following program and see the output

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    int *p, *q; /*Declaration of pointer variable*/
    int a, b; /*Declaration of variables*/
    p = &a; /*Using referencing operator to initialize pointer variable*/
    q = &b;

    system("cls");
```

```

printf("Address of a= %u\n", &a);
printf("Address of b= %u\n", &b);

printf("Value of p= %u\n", p);
printf("Value of q= %u\n", q);

printf("Enter the value of a and b:");
scanf("%d%d", &a, &b);

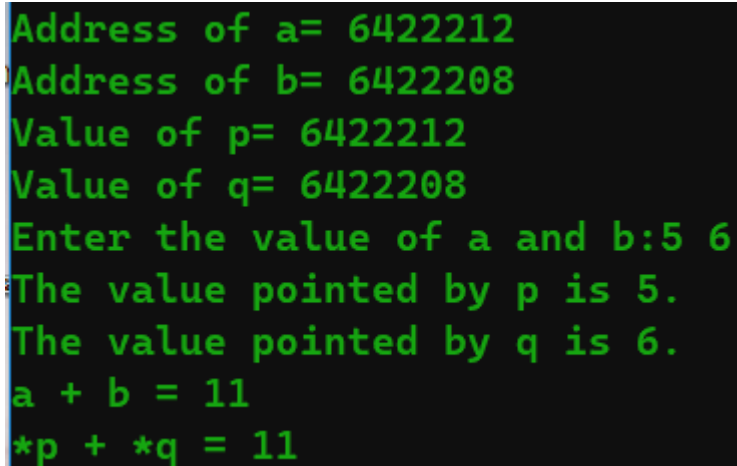
printf("The value pointed by p is %d.\n", *p); /*USing derefrencing operator*(*)*/
printf("The value pointed by q is %d.\n", *q);

printf("a + b = %d\n", a + b);
printf("*p + *q = %d\n", *p + *q);/* ----> Pointer expression*/

getch();
return 0;
}

```

Output



```

Address of a= 6422212
Address of b= 6422208
Value of p= 6422212
Value of q= 6422208
Enter the value of a and b:5 6
The value pointed by p is 5.
The value pointed by q is 6.
a + b = 11
*p + *q = 11

```

2.WAP to find the larger of two number using concept of function and pointer. Here pass two numbers function from main() to a function that finds the larger. Display the larger one from the main() function without return statement.

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

```



```

void largest(int a, int b, int *large)
{
    "Store the largest value in large variable";

    (a > b) ? (*large = a) : (*large = b);
}

int main()
{
    /*Variable Declaration*/
    int number_one, number_two, large;

    system("cls");

    /*Taking input from user*/
    printf("Number One: ");
    scanf("%d", &number_one);
    printf("Number Two: ");
    scanf("%d", &number_two);

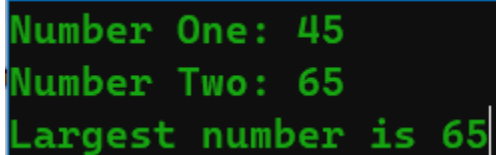
    /*Function call to find largest and store it in variable*/
    largest(number_one, number_two, &large);

    /*Printing the result*/
    printf("Largest number is %d", large);

    getch();
    return 0;
}

```

Output



```

Number One: 45
Number Two: 65
Largest number is 65

```

3.Run the following program, observe the output and comment on that Source Code

```

#include <stdio.h>

void main()

```

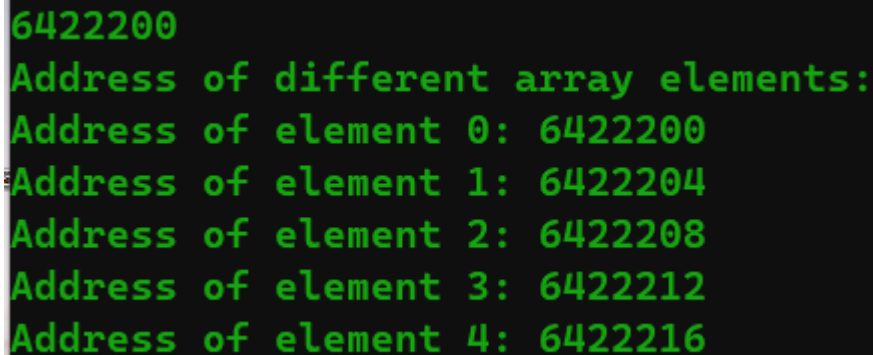
```

{
    float marks[5];
    int i;

    printf("%d\n", marks);
    printf("Address of different array elements: \n");
    for (int i = 0; i < 5; i++)
    {
        "Use either one we will obtain same result";
        /*printf("Address of element %d: %u\n", i, &marks[i]);*/
        printf("Address of element %d: %u\n", i, (marks + i));
    }
}

```

Output



```

6422200
Address of different array elements:
Address of element 0: 6422200
Address of element 1: 6422204
Address of element 2: 6422208
Address of element 3: 6422212
Address of element 4: 6422216

```

4.This program asks the required size of array to the user and displays the address of allocated blocks

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    /*Variable declaration*/
    int n, i;
    float *address; /*pointer variable declaration*/

    system("cls");

    /*Asking the size of array*/

```

```

printf("Enter the number of elements: ");
scanf("%d", &n);

/*Creating a array with DMA*/
address = (float *)calloc(n, sizeof(float)); /*Using calloc function to allocate the memory for
n number of float number.*/

/*Checking if memory is allocated or not*/
if (address == NULL)
{
    printf("Memory is not allocated.\n");
    exit(0); /*to exit the program if contents of address is NULL.*/
}

/*Printing the address of allocated blocks*/
for (i = 0; i < n; i++)
{
    printf("Address of %d block is %d.\n", i, (address + i));
}

free(address); /*To deallocate memory*/

getch();
return 0;
}

```

Output

```

Enter the number of elements: 10
Address of 0 block is 6755816.
Address of 1 block is 6755820.
Address of 2 block is 6755824.
Address of 3 block is 6755828.
Address of 4 block is 6755832.
Address of 5 block is 6755836.
Address of 6 block is 6755840.
Address of 7 block is 6755844.
Address of 8 block is 6755848.
Address of 9 block is 6755852.

```

5.1

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

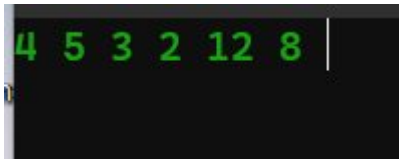
void main()
{
    int i, num[6] = {4, 5, 3, 2, 12, 8};
    int *ptr;
    ptr = num;

    system("cls");

    for (int i = 0; i < 6; i++)
    {
        printf("%d ", *(ptr + i));
    }

    getch();
}
```

Output



5.2

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    int i, num[6];
    int *ptr;
```

```

ptr = num;

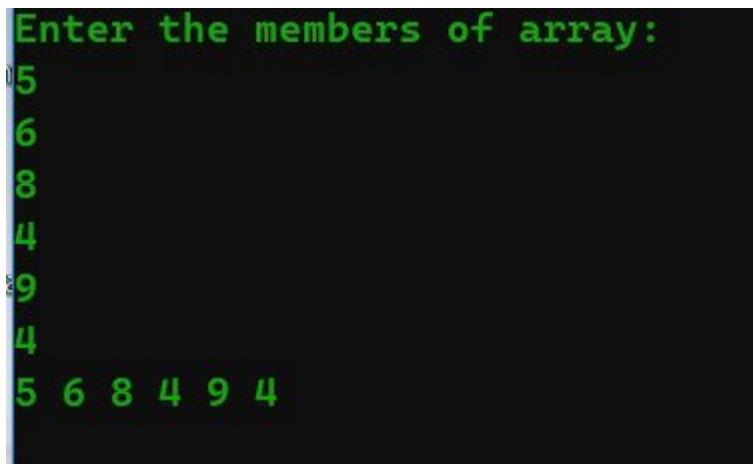
system("cls");

printf("Enter the members of array: \n");
for (int i = 0; i < 6; i++)
{
    scanf("%d", (ptr + i));
}
for (int i = 0; i < 6; i++)
{
    printf("%d ", *(ptr + i));
}

getch();
return 0;
}

```

Output



```

Enter the members of array:
5
6
8
4
9
4
5 6 8 4 9 4

```

5.3

Source Code

```
#include <stdio.h>
```

```

#include <conio.h>
#include <stdlib.h>
int main()
{
    int i, num[5], sum = 0;
    int *ptr;

    ptr = num;

    system("cls");

    printf("Enter the members of array: ");
    for (int i = 0; i < 5; i++)
    {
        scanf("%d", (ptr + i));
    }
    for (int i = 0; i < 5; i++)
    {
        (*(ptr + i) % 10 == 0 && *(ptr + i) % 15 != 0) ? (sum = *(ptr + i) + sum) : (sum =
sum);
    }

    printf("The sum of number present in the array divisible by 10 not by 15 is %d", sum);

    getch();
    return 0;
}

```

Output

```

Enter the members of array: 20
30
40
45
50
The sum of number present in the array divisible by 10 not by 15 is 110|

```

5.4 Write a program to add the elements at the corresponding position of two array of size n. Read value of n from user

Source Code

```

#include <stdio.h>

```

```

#include <conio.h>
#include <stdlib.h>

int main()
{
    /*Variable and pointer declaration*/
    int number1[100], number2[100], number[100], size;
    int *one, *two, *zero;

    /*Pointer initialization*/
    one = number1;
    two = number2;
    zero = number;

    system("cls");

    printf("Size of the array: ");
    scanf("%d", &size);

    /*Taking input*/
    printf("\nArray one:\n");
    for (int i = 0; i < size; i++)
    {
        printf("Value of %d element is ", i + 1);
        scanf("%d", (one + i));
    }

    printf("\nArray Two:\n");
    for (int i = 0; i < size; i++)
    {
        printf("Value of %d element is ", i + 1);
        scanf("%d", (two + i));
    }

    /*Calculating sum*/
    for (int i = 0; i < size; i++)
    {
        *zero = *one + *two;
        zero++;
        one++;
        two++;
    }

    //Taking pointer base address to base ko array base address
    zero -= size;

```

```

    one -= size;
    two -= size;

    /*Printing result*/
    printf("\nSum of Two Array:\n");
    for (int i = 0; i < size; i++)
    {
        printf("%d ", *(zero + i));
    }

    getch();
    return 0;
}

```

Output:

```

Size of the array: 4

Arrray one:
Value of 1 element is 5
Value of 2 element is 6
Value of 3 element is 4
Value of 4 element is 2

Array Two:
Value of 1 element is 5
Value of 2 element is 4
Value of 3 element is 8
Value of 4 element is 5

Sum of Two Array:
10 10 12 7 |

```

5.5
Source Code

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

```

```

int main()
{

```



```

int i, num[5], lowest, highest;
int *ptr;

ptr = &num[0];

system("cls");

printf("Enter the members of array: \n");
for (int i = 0; i < 5; i++)
{
    scanf("%d", (ptr + i));
}

lowest = num[0];
highest = num[0];

for (int i = 1; i < 5; i++)
{
    (lowest < *(ptr + i)) ? (lowest = lowest) : (lowest = *(ptr + i));
    (highest > *(ptr + i)) ? (highest = highest) : (highest = *(ptr + i));
}

printf("\nHighest Number is %d\n", highest);
printf("Lowest Number is %d", lowest);

getch();
return 0;
}

```

Output:

```

Enter the members of array:
45
65
12
48
45

Highest Number is 65
Lowest Number is 12

```

5.6 Write a program to read the element of array in main() pass it to function to sort the array in ascending/descending order. Display the sorted array from main().

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

void sortArray(int *num, char sort[5])
{
    int temp;
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if (strcmp(sort, "Ascend") == 0)
            {
                if (*(num + i) > *(num + j))
                {
                    {
                    }
                }
                else
                {
                    {
                        temp = *(num + i);
                        *(num + i) = *(num + j);
                        *(num + j) = temp;
                    }
                }
            }
            else if (strcmp(sort, "Descend") == 0)
            {
                {
                    if (*(num + i) < *(num + j))
                    {
                        {
                        }
                    }
                }
                else
                {
                    {
                        temp = *(num + i);
                        *(num + i) = *(num + j);
                        *(num + j) = temp;
                    }
                }
            }
        }
    }
}

int main()
```

```

{
    int i, num[5], lowest, highest;
    int *num_ptr;

    num_ptr = &num[0];
    system("cls");

    printf("Enter the members of array: ");

    for (int i = 0; i < 5; i++)
    {
        scanf("%d ", (num_ptr + i));
    }

    sortArray(num, "Ascend");
    printf("\nSorted in Ascending Order: ");

    for (int i = 0; i < 5; i++)
    {
        printf("%d ", *(num + i));
    }

    sortArray(num, "Descend");
    printf("\n\nSorted in Descending Order: ");

    for (int i = 0; i < 5; i++)
    {
        printf("%d ", *(num + i));
    }
    getch();
    return 0;
}

```

Output:

```

Enter the members of array: 4
8
2
6
9
5

Sorted in Ascending Order: 2 4 6 8 9

Sorted in Descending Order: 9 8 6 4 2

```

5.7

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    int i, num[5];
    int *num_ptr;

    num_ptr = &num[0];

    system("cls");

    printf("Enter the members of array: \n");

    for (int i = 0; i < 5; i++)
    {
        scanf("%d", (num_ptr + i));
    }

    for (int i = 0; i < 5; i++)
    {
        *(num + i) = *(num + i) * *(num + i) * *(num + i);
    }

    printf("\nNew Array: ");
    for (int i = 0; i < 5; i++)
    {
        printf("%d ", *(num + i));
    }

    getch();
    return 0;
}
```

Output:

```
Enter the members of array:
1
2
3
4
5

New Array: 1 8 27 64 125
```

5.8

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int armstrongArray(int *num)
{
    int count = 0, armstrong = 0, n, rem;

    for (int i = 0; i < 5; i++)
    {
        n = *(num + i);
        if (n == 0)
        {
            count++;
        }
        else
        {
            while (n != 0)
            {
                rem = n % 10;
                armstrong = n * n * n + armstrong;
                if (*(num + i) == armstrong)
                {
                    count++;
                }
                n = n / 10;
            }
        }
    }
}
```

```

        armstrong = 0;
    }
}

return count;
}

int main()
{
    unsigned int num[5];
    int *num_ptr;

    num_ptr = &num[0];
    system("cls");

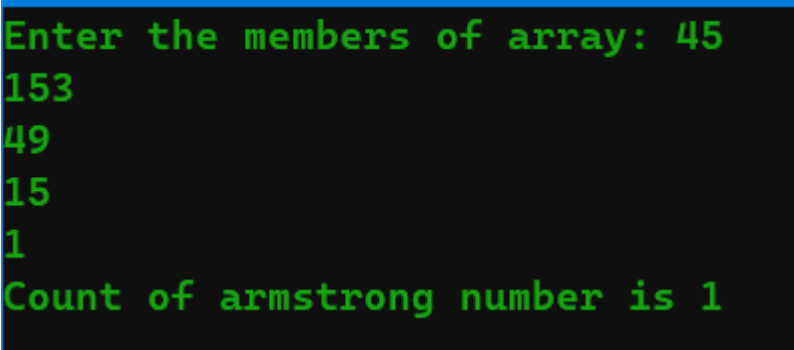
    printf("Enter the members of array: ");

    for (int i = 0; i < 5; i++)
    {
        scanf("%d", (num_ptr + i));
    }

    printf("Count of armstrong number is %d", armstrongArray(num));
    getch();
    return 0;
}

```

Output:



```

Enter the members of array: 45
153
49
15
1
Count of armstrong number is 1

```

5.9.1

Source Code

```
#include <stdio.h>
```

```
#include <conio.h>
#include <stdlib.h>
```

```
#define ARRAY_LENGTH 9
```

```
int main()
```

```
{
    int median[ARRAY_LENGTH];
    int *ptr_median;
    float result;
```

```
    ptr_median = &median[0];
```

```
    system("cls");
```

```
    for (int i = 0; i < ARRAY_LENGTH; i++)
    {
        printf("Value of %d element is ", i + 1);
        scanf("%d", (ptr_median + i));
    }
```

```
    for (int i = 0; i < ARRAY_LENGTH; i++)
    {
        for (int j = 0; j < ARRAY_LENGTH; j++)
        {
            if (*(ptr_median + i) < *(ptr_median + j))
            {
                *(ptr_median + i) = *(ptr_median + i) + *(ptr_median + j) - (*(ptr_median + j) =
                *(ptr_median + i));
            }
        }
    }
}
```

```
float position_of_array = (ARRAY_LENGTH + 1) / 2.0;
```

```
if (position_of_array - (int)position_of_array == 0.0)
{
    result = *(ptr_median + (int)position_of_array - 1);
}
else
{
    result = *(ptr_median + (int)position_of_array) + *(ptr_median + (int)position_of_array
- 1) / 2.0;
}
printf("The median of array provided is %.2f.", result);
```

```
    getch();  
    return 0;  
}
```

Output:

```
Value of 1 element is 5  
Value of 2 element is 4  
Value of 3 element is 6  
Value of 4 element is 2  
Value of 5 element is 4  
Value of 6 element is 8  
Value of 7 element is 6  
Value of 8 element is 34  
Value of 9 element is 8  
The median of array provided is 6.00.
```

5.9.2 Program to compute range of one d array

Source Code

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
  
#define ARRAY_LENGTH 8  
  
int main()  
{  
    int input[ARRAY_LENGTH], small, large, range;  
    int *ptr_input;  
  
    ptr_input = &input[0];  
  
    for (int i = 0; i < ARRAY_LENGTH; i++)  
    {  
        printf("Value of %d element is ", i + 1);
```



```

    scanf("%d", (ptr_input + i));
}

small = *(ptr_input);
large = *(ptr_input);

for (int i = 1; i < ARRAY_LENGTH; i++)
{
    if (small > *(ptr_input + i))
    {
        small = *(ptr_input + i);
        printf("small = %d\n", small);
    }
}
for (int i = 1; i < ARRAY_LENGTH; i++)
{
    if (large < *(ptr_input + i))
    {
        large = *(ptr_input + i);
    }
}
printf("large = %d\n", large);

range = (large - small);

printf("Range of given data is %d", range);

getch();
return 0;
}

```

Output:

```

Value of 1 element is 1
Value of 2 element is 5
Value of 3 element is 6
Value of 4 element is 8
Value of 5 element is 4
Value of 6 element is 2
Value of 7 element is 9
Value of 8 element is 3
large = 9
Range of given data is 8

```

5.9.3

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>

#define ARRAY_LENGTH 5

int main()
{
    int input[ARRAY_LENGTH], *ptr_input, sum = 0;
    float sd, diverse, mean = 0.0;

    ptr_input = &input[0];

    for (int i = 0; i < ARRAY_LENGTH; i++)
    {
        printf("Value of %d element is ", i + 1);
        scanf("%d", (ptr_input + i));
    }

    system("cls");

    for (int i = 0; i < ARRAY_LENGTH; i++)
    {
        sum = sum + *(ptr_input + i);
    }
    mean = sum / (float)ARRAY_LENGTH;

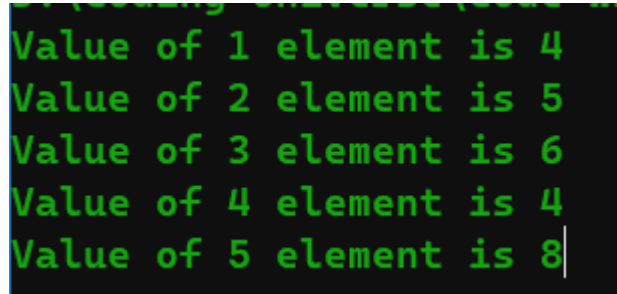
    for (int i = 0; i < ARRAY_LENGTH; i++)
    {
        diverse = (mean - *(ptr_input + i)) * (mean - *(ptr_input + i)) + diverse;
    }

    sd = pow(diverse / ARRAY_LENGTH, 0.5);

    printf("Standard Deviation is %f\n", sd);
```

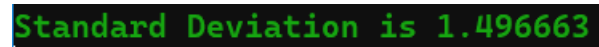
```
    getch();  
    return 0;  
}
```

Output:



A screenshot of a terminal window with a black background and green text. It displays five lines of output, each showing the value of an element in an array. The values are 4, 5, 6, 4, and 8 for elements 1 through 5 respectively. A cursor is visible at the end of the last line.

```
Value of 1 element is 4  
Value of 2 element is 5  
Value of 3 element is 6  
Value of 4 element is 4  
Value of 5 element is 8
```



A screenshot of a terminal window with a black background and green text. It displays a single line of output showing the standard deviation of the array elements.

```
Standard Deviation is 1.496663
```

5.9.4

Source Code

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
  
#define ARRAY_LENGTH 5  
  
int main()  
{  
    float variance;  
    int input[10];  
    float mean = 0.0;  
    int sum = 0;  
  
    int *ptr_input;  
  
    ptr_input = &input[0];  
  
    for (int i = 0; i < ARRAY_LENGTH; i++)  
    {  
        printf("Value of %d element is ", i + 1);  
        scanf("%d", (ptr_input + i));  
    }  
    system("cls");
```

```

for (int i = 0; i < ARRAY_LENGTH; i++)
{
    sum = sum + *(ptr_input + i);
}
mean = sum / (float)ARRAY_LENGTH;

for (int i = 0; i < ARRAY_LENGTH; i++)
{
    variance = (mean - *(ptr_input + i)) * (mean - *(ptr_input + i)) + variance;
}

variance = variance / ARRAY_LENGTH;
printf("Variance is %.2f", variance);

getch();
return 0;
}

```

Output

```

Value of 1 element is 5
Value of 2 element is 4
Value of 3 element is 6
Value of 4 element is 2
Value of 5 element is 1

```

```

Variance is 3.44

```

ANALYSIS AND DISCUSSION:

In exercise 9 of C programming, we understood about pointers, C dynamic memory allocation and their importance. The usage of pointers and functions like malloc, calloc, realloc and free were also discussed.

CONCLUSION:

Hence, the focused objective to understand the importance of pointers and dynamic memory allocation was achieved successfully.

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 10

Title: **String**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics
and

Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Strings

Code and Output

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

- In the Windows search bar, type 'settings' to open your Windows Settings.
- Search for Edit environment variables for your account.
- Choose the Path variable and then select Edit.
- Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
- Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

String

String is not a fundamental data type in C.

Like many other programming language like JavaScript or Python in C string is not a fundamental data type. In C string is a sequence of character ending in a `\0` character sequence. Thus a string with 10 character is contained in array of 11 character because there must be an `\0` character at the end for a string to exist.

If there are other characters after the `\0` character those characters are ignored and only other characters are displayed.

`\0` character indicates the end of string.

```
char a[10]={'r','a',\0,'m'};
```

Here `a` is a string with only 2 characters and `m` is not part of the string although it is part of the array.

There are a lot of string handling functions. Most of these functions accept the first argument as a pointer which represents the first element of array.

Strlen():

Syntax:

```
strlen(stringVariable)
```

is a string handling function which accepts the character pointer as its first element and returns the length of the string as output.

The important property about `strlen()` is that it looks for the first `\0` character and does not proceed further.

`strlen()` function is defined in `<string.h>` header file.
`int a= strlen(a);`

strcat():

the function prototype for `strcat()` is `char* strcat(char* dest, char * src)`

This means that the function appends the string in the `src` at the end of the `dest` and returns the `dest` pointer too.

The important precaution to be taken is `src` should be large enough to include the both first and second string.

The function adds `\0` character at the end of result but if the `dest` is small `\0` might not be included and it might end up being just array of characters.

strcpy()

`strcpy()` is a function defined in `<string.h>` to copy a string from one destination to another we can use the `strcpy()` function for this purpose,

Syntax

`char *strcpy(char *dest, const char *src)`

The function copies the string of the `src` to the string `dest` and returns the pointer to `dest`.

strcmp()

`strcmp()` function takes two strings as argument and returns the

`int strcmp(const char *str1, const char *str2)`

the function is defined in `<string.h>` and is used to analyze the two strings it returns 0 if `str1` and `str2` are completely identical

it returns a negative number if `str1` is less than `str2` and a positive number if `str1` is greater than `str2`.

1. Write the following program , observe the output and comment on it

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
```

```
void main()
{
    /*Variable Declaration and Initialization*/
    int i;
```

```
char name1[] = "pokhara city";
char name2[] = {'k', 'a', 't', 'm', 'a', 'n', 'd', 'u', '\0'};

system("cls");

/*Prints the name 1 in vertical manner*/
for (i = 0; i < strlen(name1); i++)
{
    printf("%c\n", name1[i]);
}

/*Prints the name 2 in horizontal way with a gap of one tab*/
for (i = 0; i < strlen(name2); i++)
{
    printf("%c\t", name2[i]);
}

getch();
}
```

Output:



```
p
o
k
h
a
r
a

c
i
t
y
k
    a        t        m        a        n        d        u
```

2. Write a program whether the string entered by the user is a Palindrome or not

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
    /*Variable Declaration*/
    char string[100], rev_string[100];

    system("cls");

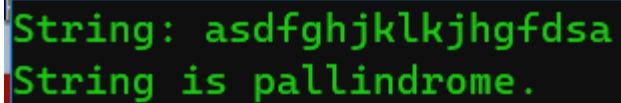
    /*Taking string*/
    printf("String: ");
    scanf("%s", string);

    /*Copy string to new variable and reversing it*/
    strcpy(rev_string, string);
    strrev(rev_string);

    /*Condition of palindrome*/
    if (strcmp(rev_string, string) == 0)
    {
        printf("String is palindrome.\n");
    }
    else
    {
        printf("String isn't palindrome.\n");
    }

    getch();
    return 0;
}
```

Output:



```
String: asdfghjklkjhgfdsa
String is pallindrome.
```

3. Write a program to read a string in main, pass it to a function that returns the count of numbers of words to main(). Display the count

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

int wordCount(char string[])
{
    "Count the words in string and return the counted words";
    int words = 0;

    for (int i = 0; i < strlen(string); i++)
    {
        if (isalpha(string[i]) && (string[i + 1] == ' ' || string[i + 1] == '\t' || string[i + 1] == '\n'))
        {
            words++;
        }
    }

    return words;
}

int main()
{
    char sentence[100];

    system("cls");

    /*Taking string*/
```

```

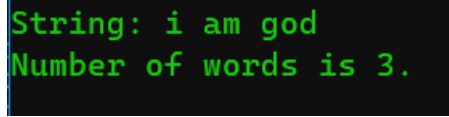
printf("String: ");
scanf("%[^\\n]", sentence);

/*Printing the result and calling function*/
printf("Number of words is %d.", wordCount(sentence));

getch();
return 0;
}

```

Output:



```

String: i am god
Number of words is 3.

```

4.TO find the highest word in the string

Source Code

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

void calculate(char sentence[10000], int *vowel, int *consonant, int *length, char
longest[100])
{
    char temp[100][20];
    int position = 0;
    int i = 0, j = 0, k = 0;

    int v = 0, c = 0, l = 0;

    while (1)
    {
        NextIteration:

        /*Count the length of words, vowel, consonant*/
        if (isalpha(sentence[j]))

```

```

{
    if (sentence[j] == 'a' || sentence[j] == 'e' || sentence[j] == 'i' || sentence[j] == 'o' ||
sentence[j] == 'u' || sentence[j] == 'A' || sentence[j] == 'E' || sentence[j] == 'I' || sentence[j]
== 'O' || sentence[j] == 'U')
    {
        v++;
        l++;
        temp[i][k] = sentence[j];
    }

    else if (sentence[j] != 'a' || sentence[j] != 'e' || sentence[j] != 'i' || sentence[j] != 'o' ||
sentence[j] != 'u' || sentence[j] != 'A' || sentence[j] != 'E' || sentence[j] != 'I' || sentence[j] !=
'O' || sentence[j] != 'U')
    {
        c++;
        l++;
        temp[i][k] = sentence[j];
    }
}

if (sentence[j] == ' ' || sentence[j] == '.' || sentence[j] == '\0')
{
    temp[i][k] = '\0';

    if (i != 0)
    {
        if (strlen(temp[position]) < strlen(temp[i]))
        {
            position = i;

            *vowel = v;
            *consonant = c;
            *length = l;
        }
    }
    if (i == 0)
    {
        *vowel = v;
        *consonant = c;
        *length = l;
    }
}

```

v = 0;

c = 0;

l = 0;

i++;

k = 0;

j++;

if (sentence[j] == '\0')

{

break;

}

goto NextIteration;

}

if (sentence[j] == '\0')

{

break;

}

j++;

k++;

}

strcpy(longest, temp[position]);

}

int main()

{

char string[200], longest[200];

int vowel, consonant, length;

system("cls");

printf("String: ");

gets(string);

calculate(string, &vowel, &consonant, &length, longest);

*printf("\nVowel = %d, \nConsonant = %d, \nLength = %d \nWords = %s\n", vowel,
consonant, length, longest);*

getch();


```
    return 0;
}
```

Output:

```
Sentence: My name is susheel thapa
Longest Word: susheel
Vowel Count: 3
Consonents Count: 7
```

5. Write a program to reverse the word using recursive function

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

void reverse(char string[], int length)
{
    "Reverse the string recursively based on length of string is even or odd";

    /*Find the length of string and it is constant so it is stored in const integer variable*/
    const int length_of_string = strlen(string);

    /*Check length is odd or even as when have seperate process of reversing odd and even length of string*/
    if (strlen(string) % 2 == 0)
    {

        /*Recursive termination condition*/
        if (length == length_of_string / 2)
        {
            return;
        }
        else
        {
            /*Swap the element of first and last position*/
            char temp = string[length_of_string - length];
            string[length_of_string - length] = string[length - 1];
            string[length - 1] = temp;
        }
    }
}
```

```

        /*Decrease length by 1*/
        length = length - 1;

        /*Function call*/
        reverse(string, length);
    }
}
else if (length_of_string % 2 == 1)
{
    /*Finding midpoint as in odd length midpoint element isn't changed*/
    const int midpoint = (length_of_string + 1) / 2;

    /*Terminating condition*/
    if (length == midpoint)
    {
        // goto stop;
        return;
    }
    else
    {
        /*Swap the last and first element*/
        char temp = string[length_of_string - length];
        string[length_of_string - length] = string[length - 1];
        string[length - 1] = temp;

        /*Decrease length by 1*/
        length = length - 1;

        /*Function call*/
        reverse(string, length);
    }
}
}

```

```

int main()
{
    /*Variable declaration*/
    char string[100];

    system("cls");

```

```

/*Taking string from user*/
printf("String: ");
scanf("%[^\\n]", string);

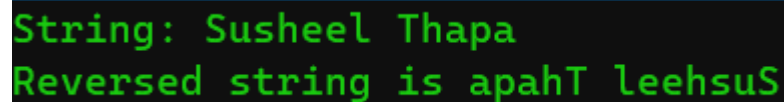
/*Function call*/
reverse(string, strlen(string));

/*Reversing the string*/
printf("Reversed string is %s\\n", string);

getch();
return 0;
}

```

Output:



```

String: Susheel Thapa
Reversed string is apahT leeHSuS

```

6. Write a separate program that exactly simulate the task of strlen(), strcat(), strcpy() and strcmp() using \user defined function

Source Code

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

/*All these function are available in string folder*/
void combine(char *str_one, char *str_two)
{
    "Combine the two string and kept combine result in first variable";

    while (*str_one != '\\0')
    {
        str_one++;
    }
}

```

```

while (*str_two != '\0')
{
    *str_one = *str_two;
    str_one++;
    str_two++;
}
*str_one = '\0';
}

```

```

int compareString(char *string_one, char *string_two)
{
    while (1)
    {
        if (*string_one != *string_two)
        {
            return 1;
        }
        else if (*string_one == *string_two)
        {
            if (*string_one == '\0' || *string_two == '\0')
            {
                return 0;
            }
        }
        if (*string_one == '\0' || *string_two == '\0')
        {
            return 1;
        }

        string_one++;
        string_two++;
    }
}

```

```

void copyString(char *destination, char *source)
{
    "Copy string from one variable to another";

    while (*source != '\0')
    {
        *destination = *source;
        destination++;
    }
}

```

```

    source++;
}

*destination = '\0';
}

int lengthString(char str[])
{
    int length = 0;
    while (str[length] != '\0')
    {
        length++;
    }
    return length;
}

int main()
{
    /*Vaeiable Declaration*/
    char string1[100], string2[100], temp[100];

    system("cls");

    /*Taking two string from user*/
    printf("String one: ");
    scanf("%s", string1);

    fflush(stdin); /*To clear buffer*/

    printf("String two: ");
    scanf("%s", string2);

    /*Copying value to temp as string2 value will be change later on*/
    copyString(temp, string2);

    /*Calling each function and printing meaningful result*/
    printf("\nUser Defined Length\n");
    printf("Length of string one is %d\n", lengthString(string1));
    printf("Length of string two is %d\n", lengthString(string2));

    printf("\nUser Defined Compare\n");
    if (compareString(string1, string2) == 0)

```

```

{
    printf("Two string is same.\n");
}
else
{
    printf("Two string isn't same.\n");
}

printf("\nUser Defined Copy\n");
copyString(string2, string1);
printf("Copying items of string one to string two: %s\n", string2);

/*Since string is change so we reinitialized it*/
copyString(string2, temp);

printf("\nUser Defined Combine\n");
combine(string1, string2);
printf("Combination of string one and string two is %s\n", string1);

getch();
return 0;
}

```

Output:

```

String one: Good Evening
String two: Good Night
:
User Defined Length
Length of string one is 12
Length of string two is 10
:
User Defined Compare
Two string isn't same.

User Defined Copy
Copying items of string one to string two: Good Evening

User Defined Combine
Combination of string one and string two is Good EveningGood Night

```

7. Write a program that reads string and arranges it in alphabetical order

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

void arrangeAlphabetical(char string[])
{
    "Arrange the string in alphabetical order. Use bubble sort algorithm";

    for (int i = 0; i < strlen(string); i++)
    {
        for (int j = 0; j < strlen(string); j++)
        {
            /*Below if will convert first character to lower and then compare so the ascending order is arranged*/
            if (((int)tolower(string[i]) - (int)tolower(string[j])) < 0)
            {
                char temp = string[i];
                string[i] = string[j];
                string[j] = temp;
            }
        }
    }
}

int main()
{
    /*Variable Declaration*/
    char string[100];

    system("cls");

    /*Taking input*/
    printf("String: ");
    scanf("%s", string);

    /*Arranging String*/
    arrangeAlphabetical(string);
}
```

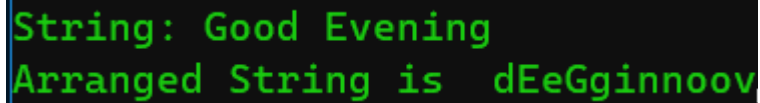
```

    /*Printing result*/
    printf("Arranged String is %s", string);

    getch();
    return 0;
}

```

Output:



```

String: Good Evening
Arranged String is dEeGginnoov

```

8. Write a program to find the frequency of the character in the string entered by the user

Source Code

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    char string[100];
    char ch;
    int i = 0, count = 0;

    system("cls");

    printf("Enter the string:: ");
    gets(string);

    printf("Character Frequency:: ");
    scanf("%c", &ch);

    while (string[i] != '\0')
    {
        if (string[i] == ch)
        {
            count++;
        }
    }
}

```



```

        i++;
    }

    printf("Frequency of the character %c is %d.", ch, count);

    getch();
    return 0;
}

```

Output

```

Enter the string:: He is very good boy lives in himalyan region
Character Frequency:: a
Frequency of the character a is 2.

```

9. Write a program to find the frequency of the characters in the string enter by the user

Source Code

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

struct Frequency
{
    char alphabet;
    int count;
} Character[26];

int main()
{
    char string[100];
    int i = 0, number_of_character = 0;

    system("cls");

    printf("Enter the string::\n");
    gets(string);

```

```

while (string[i] != '\0')
{
    if (i == 0)
    {
        Character[0].alphabet = string[i];
        Character[0].count = 1;
        number_of_character++;
        goto Last;
    }

    for (int j = 0; j < number_of_character; j++)
    {
        if (string[i] == Character[j].alphabet)
        {
            Character[j].count++;
            goto Last;
        }
    }

    Character[number_of_character].alphabet = string[i];
    Character[number_of_character].count = 1;
    number_of_character++;
Last:
    i++;
}

printf("%-15s%-15s\n", "Character", "Frequency");

for (i = 0; i < number_of_character; i++)
{
    printf("%-15c%-15d\n", Character[i].alphabet, Character[i].count);
}

getch();
return 0;
}

```

Output:

```
Enter the string::
He lives in Nepaljung
Character      Frequency
H              1
e              3
l              3
i              2
v              2
s              1
n              1
N              2
p              1
a              1
j              1
u              1
g              1
```

10. Write a program to read a string in main(). Pass it to function. The function convert all the uppercase of lower and vice-verse

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

void convertLowerUpper(char *string)
{
    "Convert the lower case to uppercase and vice verse";

    while (*string != '\0')
    {
```

```

        (isupper(*string)) ? (*string = tolower(*string)) : (*string = toupper(*string));
        string++;
    }
}

int main()
{
    /*Variable Declaration*/
    char string[100];

    system("cls");

    /*Taking string input*/
    printf("String: ");
    scanf("%[^\\n]", string);

    /*Function call*/
    convertLowerUpper(string);

    /*Printing result*/
    printf("\\nConverted String : %s\\n", string);

    getch();
    return 0;
}

```

Output:

```

String: Susheel is vnajASDJFva sdfja Adjf
Converted String : sUSHEEL IS VNAJasdjfVA SDFJA aDJF

```

11. Write a program to read the name of 10 students in main(). Pass the name list to function that sorts the array to ascending order. Display the sorted array from main().
Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

void sort(char name[][100])
{
    "Compare each string and arrange in alphabetical order. BUBBLE SORT TECHNIQUE";

    char temp[100];
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            if (strcmpi(name[i], name[j]) < 0)
            {
                strcpy(temp, name[i]);
                strcpy(name[i], name[j]);
                strcpy(name[j], temp);
            }
        }
    }
}

int main()
{
    /*Variable Declaration*/
    char student_name[10][100];

    system("cls");

    /*Taking Input*/
    for (int i = 0; i < 10; i++)
    {
        fflush(stdin);

        printf("Name: ");
        scanf("%[^\\n]", student_name[i]);
    }
}
```

```
}

/*Function call*/
sort(student_name);

/*Printing Result*/
printf("\nName in Alphabetical Order\n");
for (int i = 0; i < 10; i++)
{
    printf("%s\n", student_name[i]);
}

getch();
return 0;
}
```

Output:

```
Name: Hari
Name: Sita
Name: Gita
Name: Rita
Name: Ram
Name: Shyam
Name: Krishna
Name: Bisnu
Name: Susheel
Name: Babita

Name in Alphabetical Order
Babita
Bisnu
Gita
Hari
Krishna
Ram
Rita
Shyam
Sita
Susheel
```

12

Write a program to do following things:

-->To print the question "Who is prime minister of Nepal?"

-->To accept answer

-->TO print "Good" and stop if the answer is correct

-->To print the message "try again", if the answer is wrong.

-->To display the correct answer when the answer is wrong even at the third attempt,

exit the program

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
    /*Variable Declaration*/
    char answer[50];
    int attempt = 0;

    system("cls");

    /*While loop to take,check answer and print infomation on the answer and stop when
attempt is 3*/
    while (attempt != 3)
    {
        fflush(stdin); /*CLear Buffer*/

        /*Asking Question and receivng answer*/
        printf("Who is prime minister of Nepal?\n");
        scanf("%[^\\n]", answer);

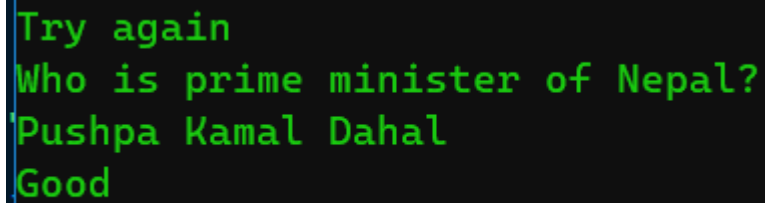
        /*Checking Answer and taking decision*/
        if (strcmpi(answer, "Pushpa Kamal Dahal") == 0)
        {
            printf("Good");
            break;
        }
        else
        {
            system("cls");
```

```
        printf("Try again\n");
        attempt++;
    }
}

/*If user gives all wrong answer*/
if (attempt == 3)
{
    printf("\nYou have tried for 3 times but you were unable to give correct answer.\n");
    printf("\nCorrect Answer: %s", "Pushpa Kamal Dahal");
}

getch();
return 0;
}
```

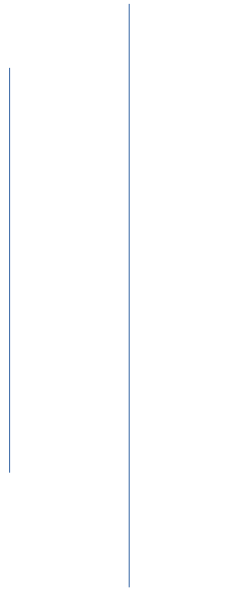
Output:

A screenshot of a terminal window with a black background and green text. The text shows the program's execution: it prompts the user to try again, asks for the prime minister of Nepal, receives the correct answer 'Pushpa Kamal Dahal', and ends with 'Good'.

```
Try again
Who is prime minister of Nepal?
Pushpa Kamal Dahal
Good
```


INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 11

Title: **Structures**

Submitted by:
Susheel Thapa 077BCT090

Submitted to:
Department of Electronics and
Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Structures

Code and Output (Analysis)

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#). Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows](#) from SourceForge.net. Your download will start automatically. Run the downloaded .exe file. After, you have install MinGW-w64, you need to configure it.

1. In the Windows search bar, type 'settings' to open your Windows Settings.
2. Search for Edit environment variables for your account.
3. Choose the Path variable and then select Edit.
4. Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
5. Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

Structure:

It is a heterogeneous user defined data type. It is also called constructed data type. It may contain different data types. Structure can also store non homogenous data type into a single collection. Structure may contain pointer, arrays, or even other structures other than the common data types such as int, float, long int etc. A structure provides a means of grouping variables under a single name for easier handling and identification. It can be defined as new named types. It is a convenient way of grouping several pieces of related information together. Complex hierarchies can be created by nesting structures. Structures may be copied to and assigned. They are also useful in passing groups of logically related data into structures. The declaration of structures is given below:

struct tag

```
{
    member 1;
    member 2;

    member n;
};
```

1. Write a program to create a structure having members: Name, Address, Telephone number and Salary of an employee. Read the values of members from the user and display.

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

/*Creating structure*/
struct UserData
{
    /*Member of structure*/
    char user_name[100];
    char user_address[100];
    long long int user_telephone_number;
    long int user_salary;
};

int main()
{
    /*Creating Structure Variable*/
    struct UserData UD_one;

    system("cls");
```

```

/*Taking input from user*/
printf("User Name: ");
scanf("%s", UD_one.user_name);

printf("User Address: ");
scanf("%s", UD_one.user_address);

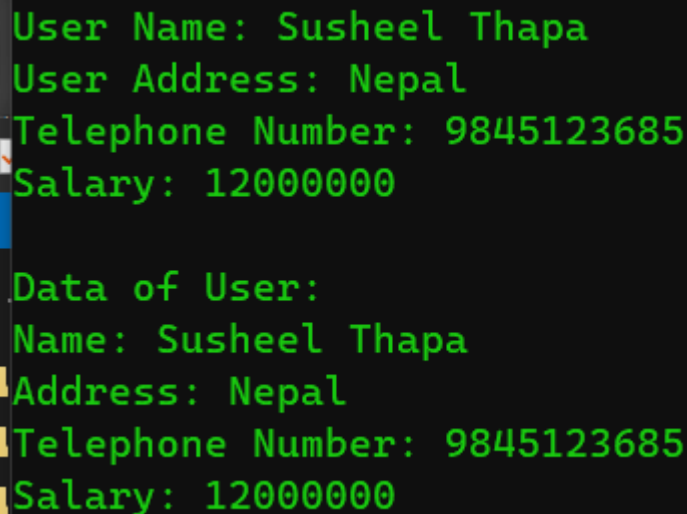
printf("Telephone Number: ");
scanf("%ld", &UD_one.user_telephone_number);

printf("Salary: ");
scanf("%ld", &UD_one.user_salary);

/*Print the taken value*/
printf("\nData of User:");
printf("\nName: %s", UD_one.user_name);
printf("\nAddress: %s", UD_one.user_address);
printf("\nTelephone Number: %ld", UD_one.user_telephone_number);
printf("\nSalary: %ld", UD_one.user_salary);
getch();
return 0;
}

```

Output:



```

User Name: Susheel Thapa
User Address: Nepal
Telephone Number: 9845123685
Salary: 12000000

Data of User:
Name: Susheel Thapa
Address: Nepal
Telephone Number: 9845123685
Salary: 12000000

```

2.Create a structure employee containing names as character string , telephone as character string and salary as intger. Input the records of 10 employees. After that display the name,telephone and salary of employee with highest salary and lowest salary and display the average salary of all 10 employee

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

struct Employee
{
    /*Member of structure*/
    char name[100];
    char telephone_number[100];
    int salary;
};

int main()
{
    struct Employee e[10];
    int highest_salary, lowest_salary;

    float average_salary = 0;

    system("cls");

    /*Taking input from user*/
    for (int i = 0; i < 10; i++)
    {
        printf("\nEmployee %d\n", i + 1);
        printf("Name: ");
        scanf(" %[^\\n]", e[i].name);

        printf("Telephone Number: ");
        scanf(" %[^\\n]", e[i].telephone_number);

        printf("Salary: ");
        scanf("%ld", &e[i].salary);
    }

    highest_salary = e[0].salary;
    lowest_salary = e[0].salary;

    system("cls");

    for (int i = 0; i < 10; i++)
    {
```

```

    (highest_salary < e[i].salary) ? (highest_salary = e[i].salary) : (highest_salary =
highest_salary);
    (lowest_salary > e[i].salary) ? (lowest_salary = e[i].salary) : (lowest_salary =
lowest_salary);

    average_salary = (average_salary * i + e[i].salary) / (i + 1);
}

for (int i = 0; i < 10; i++)
{
    if (e[i].salary == highest_salary)
    {
        printf("\n<---Highest Salary Employee---->\n");
        printf("Name: %s\n", e[i].name);
        printf("Telephone Number: %s\n", e[i].telephone_number);
        printf("Salary: %d\n", e[i].salary);
    }
    if (e[i].salary == lowest_salary)
    {
        printf("\n<---Lowest Salary Employee---->\n");
        printf("Name: %s\n", e[i].name);
        printf("Telephone Number: %s\n", e[i].telephone_number);
        printf("Salary: %d\n", e[i].salary);
    }
}

printf("\nAverage Salary of the employee: %f\n", average_salary);

getch();
return 0;
}

```

Output:

```
<---Highest Salary Employee--->
Name: G
Telephone Number: 130
Salary: 909

<---Lowest Salary Employee--->
Name: I
Telephone Number: 156
Salary: 127

Average Salary of the employee: 573.000000
```

**2.Create a structure Data containing three members: int dd,int mm,int yy.
Create another member date of birth, create an object of structure data inside person.
Using these structure, Write a program to input records until user enter 'n' or 'N'.Then,
display the contents in tabular form.**

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

/*A Structure*/
struct Date
{
    int dd;
    int mm;
    int yy;
};

/*Nested Structure*/
struct Person
{
    char name[100];
    char address[100];
    long long int telephone_number;
    struct Date date_of_birth;
};
```



```

int main()
{
    /*Creating the variable of srtucture*/
    struct Person p[100];
    int records = 0;

    system("cls");

    /*While loop to take data for infinite times till user enter n */
    while (1)
    {
        printf("Name: ");
        scanf(" %[^\\n]", p[records].name);

        if (strcmp(p[records].name, "n") == 0)
        {
            break;
        }

        printf("Address: ");
        scanf(" %[^\\n]", p[records].address);

        printf("Telephone Number: ");
        scanf("%lld", &p[records].telephone_number);

        printf("Date of Birth:\\n");
        printf(" Day: ");
        scanf("%d", &p[records].date_of_birth.dd);
        printf(" Month: ");
        scanf("%d", &p[records].date_of_birth.mm);
        printf(" Year: ");
        scanf("%d", &p[records].date_of_birth.yy);

        records++;
    }

    system("cls");

    /*Printing the data in tabular way*/
    printf("%10s%30s%30s%20s\\n", "DOB", "Name", "Address", "Telephone Number");
    for (int i = 0; i < records; i++)
    {
        printf("%02d-%02d-%04d%30s%30s%20lld", p[i].date_of_birth.dd, p[i].date_of_birth.mm,
p[i].date_of_birth.yy, p[i].name, p[i].address, p[i].telephone_number);
        printf("\\n");
    }
}

```

```

    }

    getch();
    return 0;
}

```

Output:

DOB	Name	Address	Telephone Number
14-02-2002	Susheel Thapa	Chitwan	9811111111
26-24-2003	Rajiv Chaudary	Parsa	98156489978
09-12-2000	Shree Thapa	Nipani	9845781245

4.Create a structure TIME containing hour, minutes and seconds as its member. Write a program that uses this structure to input start time and stop time in main(). Pass the structure to a function that calculated the sum and diffrencece of start time and stop time.Display the sum and difference from main()

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

/*Creating Structure*/
struct TIME
{
    int hh;
    int mm;
    int ss;
};

/*Function with type of created structure*/
struct TIME addTime(struct TIME first, struct TIME second)
{
    "Add the two time passed";

    struct TIME result;

    result.ss = first.ss + second.ss;
    result.mm = first.mm + second.mm;
}

```

```

result.hh = first.hh + second.hh;

if (result.ss >= 60)
{
    result.ss = result.ss - 60;
    result.mm = result.mm + 1;
}
if (result.mm >= 60)
{
    result.mm = result.mm - 60;
    result.hh = result.hh + 1;
}
return result;
}

struct TIME subtractTime(struct TIME first, struct TIME second)
{
    "Subtract the two time passes";

    struct TIME result;

    result.ss = first.ss - second.ss;
    result.mm = first.mm - second.mm;
    result.hh = first.hh - second.hh;

    if (result.ss < 0)
    {
        result.ss = result.ss + 60;
        result.mm = result.mm - 1;
    }
    if (result.mm < 0)
    {
        result.mm = result.mm + 60;
        result.hh = result.hh - 1;
    }
    return result;
}

int main()
{
    struct TIME start, stop, add, subtract;

    system("cls");

    /*Taking input*/

```

```
printf("Start Time:\n");
printf("Hour: ");
scanf("%d", &start.hh);
printf("Minutes: ");
scanf("%d", &start.mm);
printf("Seconds: ");
scanf("%d", &start.ss);
```

```
printf("\nStop Time:\n");
printf("Hour: ");
scanf("%d", &stop.hh);
printf("Minutes: ");
scanf("%d", &stop.mm);
printf("Seconds: ");
scanf("%d", &stop.ss);
```

```
/*Function call*/
add = addTime(start, stop);
subtract = subtractTime(start, stop);
```

```
/*Printing result*/
printf("Added Times(hh:mm:ss)\n");
printf("%d:%d:%d\n", add.hh, add.mm, add.ss);
```

```
printf("\nSubtracted Times(hh:mm:ss)\n");
printf("%d:%d:%d\n", subtract.hh, subtract.mm, subtract.ss);
```

```
getch();
return 0;
```

```
}
```

Output:

```
Start Time:
Hour: 4
Minutes: 56
Seconds: 23

Stop Time:
Hour: 3
Minutes: 15
Seconds: 20
Added Times(hh:mm:ss)
8:11:43

Subtracted Times(hh:mm:ss)
1:41:3
```

5. Write a program to compute any two instant memory spaces in format (Kilobytes:Bytes:Bits) using structure. Build function to add and subtract given memory spaces where 1kb = 1-24b and 1b = 8bits and display the result from main()

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

/*Creating Structure*/
struct Memory
{
    int kb;
    int b;
    int bits;
};
```

```

/*Function with type of created structure*/
struct Memory addMemory(struct Memory first, struct Memory second)
{
    "Add the two memory passed";

    struct Memory result;

    result.bits = first.bits + second.bits;
    result.b = first.b + second.b;
    result.kb = first.kb + second.kb;

    if (result.bits >= 8)
    {
        result.bits = result.bits - 8;
        result.b = result.b + 1;
    }
    if (result.b >= 1024)
    {
        result.b = result.b - 1024;
        result.kb = result.kb + 1;
    }
    return result;
}

struct Memory subtractMemory(struct Memory first, struct Memory second)
{
    "Subtract the two Memory passed";

    struct Memory result;

    result.bits = first.bits - second.bits;
    result.b = first.b - second.b;
    result.kb = first.kb - second.kb;

    if (result.bits < 0)
    {
        result.bits = result.bits + 8;
        result.b = result.b - 1;
    }
    if (result.b < 0)
    {
        result.b = result.b + 1024;
        result.kb = result.kb - 1;
    }
    return result;
}

```

```

}

int main()
{
    struct Memory first, second, add, subtract;

    system("cls");

    /*Taking input*/
    printf("First Memory:\n");
    printf("Kilobytes: ");
    scanf("%d", &first.kb);
    printf("Bytes: ");
    scanf("%d", &first.b);
    printf("Bits: ");
    scanf("%d", &first.bits);

    printf("\nSecond Memory:\n");
    printf("Kilobytes: ");
    scanf("%d", &second.kb);
    printf("Bytes: ");
    scanf("%d", &second.b);
    printf("Bits: ");
    scanf("%d", &second.bits);

    /*Function call*/
    add = addMemory(first, second);
    subtract = subtractMemory(first, second);

    /*Printing result*/
    printf("Added Memory(kb:b:bits)\n");
    printf("%d:%d:%d\n", add.kb, add.b, add.bits);

    printf("\nSubtracted Memory(kb:b:bits)\n");
    printf("%d:%d:%d\n", subtract.kb, subtract.b, subtract.bits);

    getch();
    return 0;
}

```

Output:

```
First Memory:
Kilobytes: 15
Bytes: 4
Bits: 2

Second Memory:
Kilobytes: 14
Bytes: 5
Bits: 4
Added Memory(kb:b:bits)
29:9:6

Subtracted Memory(kb:b:bits)
0:1022:6
```

Analysis

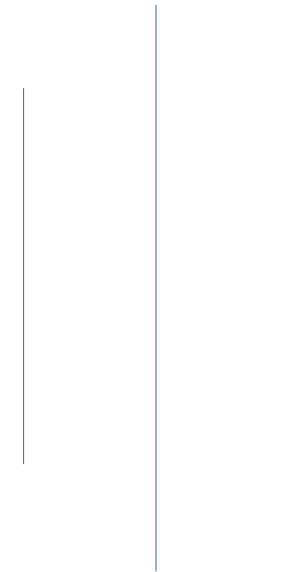
We learn about how we can declare new data types, use it , take data and print out the taken data and many more.

Conclusion:

From this we have learn about structures

INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 12

Title: **Files Handling**

Submitted by:
Susheel Thapa 077BCT090

Submitted to:
Department of Electronics and
Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Files

Code and Output (Analysis)

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#). Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows](#) from SourceForge.net. Your download will start automatically. Run the downloaded .exe file. After, you have install MinGW-w64, you need to configure it.

1. In the Windows search bar, type 'settings' to open your Windows Settings.
2. Search for Edit environment variables for your account.
3. Choose the Path variable and then select Edit.
4. Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
5. Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

File:

Many applications require that information be written to or read from an auxiliary memory device. Such information is stored on the memory device in the form of a data file. The data files allow us to store information permanently and to access and alter that information whenever necessary.

Opening a file:

Before performing any input / output operation, file must be opened. While opening file, the following must be specified:-

- i. The name of file.
- ii. The manner in which it should be opened (that for reading ,writing ,both reading and writing ,appending at the end of file, overwriting the file)
- iii. when working with a stream oriented data file ,the first step is to establish a buffer area, where information is temporary stored while being transferred between the computers memory and data file .the buffer area is established by writing
`FILE *fpt;`

where File is a special structure type establishes the buffer area and ptvar is a pointer variable that indicates the beginning of the buffer area the library function `fopen` is used to open a file .This function is used to open a file .This function is typically written as

`fpt=fopen(file name, file type);`

where file name and file type are strings that represent the name of the data file and the manner in which the data file will be utilized.

Finally, a file can be closed at the end of the program. This can be accomplished with the library function `fclose`. The syntax is simply,
`fclose(fpt)`

Processing a file:

Most data file application requires that a data file be altered as it is being processed. For example, in an application involving the processing of customer records, it may be desirable to add new records to the file. To delete the existing records, to modify the contents or to rearrange the records.

File types:

DOS treats files in two different ways that as binary or text file. Files almost all UNIX systems do not make any distinction between the two. If a file is specified as the binary type, the file I /O functions do not interpret the contents of the file when they read from, or write to the file. But if the file is specified as the text type, the file I / O functions interpret the contents of the file. The basic differences in these two types of files are:

- i. ASCII 0*1a is considered as end of file character by the file I /O functions when reading from a text file and they assume that the end of the file has been reached.
- ii. In case of DOS, a new line is stored as the sequence 0*0a on the disk in case text files.

1. Write a program to read a year from the user and write it in the file if the year is leap year

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

void leapYear(unsigned int year)
{

    unsigned int divisible_by_four, divisible_by_hundered, divisible_by_four_hundered;

    /*Below if check whether the year enter in after 1582 or not*/
    if (year >= 1582)
    {

        /*Needed tool for checking the variable*/
        divisible_by_four = year % 4;
        divisible_by_hundered = year % 100;
        divisible_by_four_hundered = year % 400;

        /*Check the condition to be a leap year*/
        if (divisible_by_four == 0)
        {
            if (!(divisible_by_hundered == 0) || divisible_by_four_hundered == 0)
            {
                printf("It is leap year.\n");

                printf("Writing to file 'Program01.txt'.\n");

                /*Writing data to files*/
                FILE *ptr_file; /*Files pointer*/

                ptr_file = fopen("Program01.txt", "a"); /*Pointing to the file*/
```

```

fprintf(ptr_file, "%d is leap year.\n", year);

fclose(ptr_file); /*Close the files pointer*/
printf("\n\nDONE...");
}
else
{
printf("It isn't leap year.");
}
}
else
{
printf("It isn't leap year.");
}
}
else
{
printf("You didn't give input as per asked in questions.");
}
}
}
int main()
{
    unsigned int year;

    system("cls");

    printf("Year(after 1582): ");
    scanf("%u", &year);

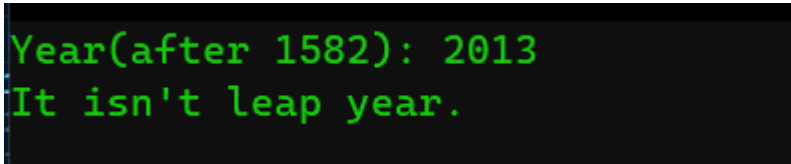
    leapYear(year);

    getch();
    return 0;
}

```

Output:

Screenshot of only one output



```

Year(after 1582): 2013
It isn't leap year.

```

Files data with various input.

```
2000 is leap year.  
2004 is leap year.  
2000 is leap year.
```

2.WAP to read words from the user until user hits 'NO' and write them to a file if the word is vowel free. Display the content in the files

Source Code:

```
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
#include <string.h>  
  
int vowelCheck(char *word)  
{  
    "Check whether the word contain the vowel letter of not";  
  
    char vowel[] = "aeiouAEIOU";  
  
    while (*word != '\0')  
    {  
        for (int i = 0; i < 10; i++)  
        {  
            if (*word == vowel[i])  
            {  
                return 1;  
            }  
        }  
        word++;  
    }  
    return 0;  
}  
  
int main()
```

```

{
    char word[100], file_content;
    FILE *ptr_word;
    int vowel_check_result;

    system("cls");

    do
    {
        printf("Words: ");
        scanf("%s", word);

        vowel_check_result = vowelCheck(word);

        if (vowel_check_result == 0)
        {
            ptr_word = fopen("Program02.txt", "a");

            fprintf(ptr_word, "~%s\n", word);

            fclose(ptr_word);
        }
    } while (strcmpi(word, "NO") != 0);

    /*Reading and printing the content of file*/
    printf("Content of file:\n");
    ptr_word = fopen("Program02.txt", "r");
    do
    {
        file_content = fgetc(ptr_word);

        printf("%c", file_content);
    } while (file_content != EOF);

    fclose(ptr_word);

    /*Deleting the content of files*/
    ptr_word = fopen("Program02.txt", "w");
    fclose(ptr_word);

    getch();
    return 0;
}

```


Output:

```
Words: sus
Words: he
Words: is
Words: whnn
Words: uy
Words: re
Words: no
Content of file:
~whnn
```

4. Write a program to open a new file, read roll number, name, address and phone number of students until user says "no" after reading the data write it to file then display the contents of file

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
    FILE *in, *out;
    char roll_number[100], name[100], address[100];
    long long int phone_number;

    // system("cls");
    in = fopen("Program03.txt", "a");
```

```

while (1)
{
// fflush(stdin);

printf("Roll Number: ");
scanf(" %[^'\n']", roll_number);

// printf("%s", roll_number);
// exit(0);

if (strcmpi(roll_number, "no") == 0)
{
break;
}

printf("Name: ");
scanf(" %[^'\n']", name);

    printf("Address: ");
    scanf(" %[^'\n']", address);

    printf("Phone Number: ");
    scanf("%lld", &phone_number);
    fprintf(in, "%30s%30s%30s%15lld\n", roll_number, name, address, phone_number);
}

fclose(in);

system("cls");

out = fopen("Program03.txt", "r");

printf("\n%30s%30s%30s%15s\n", "Roll Number", "Name", "Address", "Phone Number");

while (1)
{
    char ch = fgetc(out);
    if (ch == EOF)
    {
        break;
    }
    printf("%c", ch);
}
fclose(out);

```

```

    getch();
    return 0;
}

```

Output:

Roll Number	Name	Address	Phone Number
1	Ram	Chitwan	9800000000
2	Shyam	Lamjung	9811111111
3	Hari	Nepaljung	9822222222
4	Geeta	Hetauda	9833333333
5	Sita	Kathmandu	9844444444
6	Balram	Sagarmatha	9855555555
7	Hanuman	Parbat	9866666666
8	EarthQuake	Nepal	9877777777
9	Laxman	Jhapa	9888888888
090	Susheel	Bharatpur-05, Chitwan	9816156426
023	Something	Parsa	9811564794
12	Noone	Heaven	9874562135

Data in files

1	Ram	Chitwan	9800000000
2	Shyam	Lamjung	9811111111
3	Hari	Nepaljung	9822222222
4	Geeta	Hetauda	9833333333
5	Sita	Kathmandu	9844444444
6	Balram	Sagarmatha	9855555555
7	Hanuman	Parbat	9866666666
8	EarthQuake	Nepal	9877777777
9	Laxman	Jhapa	9888888888
090	Susheel	Bharatpur-05, Chitwan	9816156426
023	Something	Parsa	9811564794
12	Noone	Heaven	9874562135

4. Write a program to input name, roll, address, telephone number and score of a student. Store the contents of the person in file first.txt After that, copy the contents of first.txt to second.txt and display the contents of second.txt In this program, you should use the text

files.[You can use structure for data handling]

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

struct Data
{
    char name[100];
    int roll_number;
    char address[100];
    long long int telephone_number;
    float score;
};

int main()
{
    char ch;

    /*Structure variable*/
    struct Data user;

    /*Creating Files pointer*/
    FILE *in_first;
    FILE *out_first;
    FILE *in_second;
    FILE *out_second;

    system("cls");

    /*Taking input from user*/
    printf("Name: ");
    scanf("%s", user.name);

    printf("Roll Number: ");
    scanf("%d", &user.roll_number);

    printf("Address: ");
    scanf("%s", user.address);

    printf("Telephone Number:");
```

```

scanf("%lld", &user.telephone_number);

printf("IOE score: ");
scanf("%f", &user.score);

/*Opening first.txt in write mode and write the inputted data*/
in_first = fopen("First.txt", "w");
fprintf(in_first, "%s %d %s %lld %.2f", user.name, user.roll_number, user.address,
user.telephone_number, user.score);
fclose(in_first);

/*Opening First.txt in read mode and Second.txt in write mode*/
out_first = fopen("First.txt", "r");
in_second = fopen("Second.txt", "w");

/*Coping character from first.txt to second.txt character by character*/
while (1)
{

    /*Taking a character*/
    ch = fgetc(out_first);

    /*Condition if the character in the files has reached to end*/
    if (ch == EOF)
    {
        break;
    }
    fputc(ch, in_second); /*First argument character , Second argument File pointer*/
}

/*Closing the files*/
fclose(out_first);
fclose(in_second);

/*Opening the second.txt in read mode and reading charcter by character and printing*/
out_second = fopen("Second.txt", "r");
while (1)
{
    ch = fgetc(out_second);
    if (ch == EOF)
    {
        break;
    }
    printf("%c", ch);
}

```

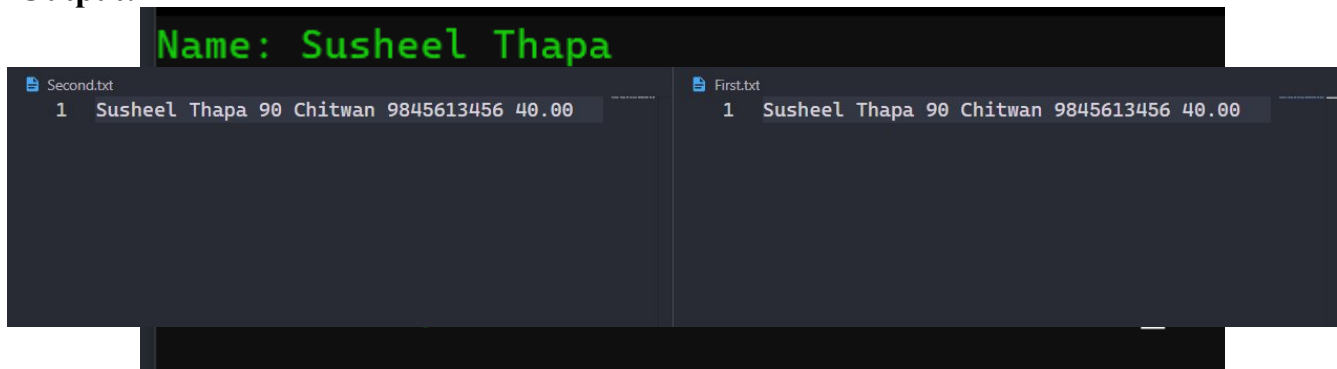
```

fclose(out_second);

getch();
return 0;
}

```

Output:



5. Modify question number 4, using binary file. You should use `fwrite()` function to write data into a text file and `fread()` function to read the file. [here, you must use structure for data handling]

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

/*Creating a structure*/
struct Data
{
    char name[100];
    int roll_number;
    char address[100];
    long long int telephone_number;
    float score;
};

int main()
{
    /*Structure variable*/

```

```

struct Data user;

/*Creating Files pointer*/
FILE *in_first;
FILE *out_first;
FILE *in_second;
FILE *out_second;

system("cls");

/*Taking input from user*/
printf("Name: ");
scanf("%s", user.name);

printf("Roll Number: ");
scanf("%d", &user.roll_number);

printf("Address: ");
scanf("%s", user.address);

printf("Telephone Number:");
scanf("%lld", &user.telephone_number);

printf("IOE score: ");
scanf("%f", &user.score);

/*Opening first.txt in write binary mode and write the inputted data*/
in_first = fopen("First_05.txt", "wb");

/*Writing data that has been taken from user*/
fwrite(&user, sizeof(user), 1, in_first);

/*Closing the file*/
fclose(in_first);

/*Opening First.txt in read binary mode and Second.txt in write binary mode*/
out_first = fopen("First_05.txt", "rb");
in_second = fopen("Second_05.txt", "wb");

/*Reading a data and writing the read data to another file. Copying data from one file to
another*/
fread(&user, sizeof(user), 1, out_first);
fwrite(&user, sizeof(user), 1, in_second);

/*Closing the files*/

```

```

fclose(out_first);
fclose(in_second);

/*Opening the second.txt in read binary mode*/
out_second = fopen("Second_05.txt", "rb");

/*Reading the data*/
fread(&user, sizeof(user), 1, out_first);

/*Printing the data that has been read*/
printf("%s\n", user.name);
printf("%s\n", user.address);
printf("%lld\n", user.telephone_number);
printf("%d\n", user.roll_number);
printf("%.2f\n", user.score);
fclose(out_second);

getch();
return 0;
}

```

Output:

```

Name: Susheel Thapa
Roll Number: 156
Address: Bharatpur-05, Chitwan
Telephone Number:9845612378
IOE score: 50
Susheel Thapa
Bharatpur-05, Chitwan
9845612378
156
50.00

```


Analysis and Discussion

Here we have written a lot of programs and run it many times. Through this, we are able to know the respective syntax of the element like `fprintf()`, `fscanf()`, `fgets ()`, `fgetc()`, `fputc()` and many more.

Moreover, it provides us brief information about how can we write into files read into files and many more.

Conclusion

In nut shell we learn about file handling