Problem Solving

Difficulty Level: Easy

Problems:

1. Check Whether a Given Number is Even or Odd

   Even Number:
   A number is said to be an even number if it is completely divisible by 2.
   In other words, if a number is divided by 2 and leaves a remainder of 0, then it is said to be an even number.
   Example: 36, 24

   Odd Number:
   A number is said to be an odd number if it is not completely divisible by 2.
   In other words, if a number is divided by 2 and the remainder is 1, it is said to be an odd number.
   Example: 21, 15

   Program :

```
#include <stdio.h>

int main() {

int num;

printf("Enter an integer: ");

scanf("%d", &num);



if(num % 2 == 0)

printf("%d is even.", num);

else

printf("%d is odd.", num);



return 0;

}
```

2. Find the Sum of Even and Odd Numbers
   Problem Description:
   The program takes the number N and finds the sum of odd and even numbers from 1 to N.

   Program :

```
#include<stdio.h>

int main()
{
int i, number, Even_Sum = 0, Odd_Sum = 0;

printf("\n Please Enter the Maximum Limit Value : ");
scanf("%d", &number);

for(i = 1; i <= number; i++)
{
if ( i%2 == 0 )
{
Even_Sum = Even_Sum + i;
}
else
{
Odd_Sum = Odd_Sum + i;
}
}
printf("\n The Sum of Even Numbers upto %d  = %d", number, Even_Sum);
printf("\n The Sum of Odd Numbers upto %d  = %d", number, Odd_Sum);

return 0;
}
```

3. Check Whether a Number is Positive or Negative
   Problem Description:
   The program takes the given integer and checks whether the integer is positive or negative.

   Program :

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

```
int main() {

double num;
printf("Enter a number: ");
scanf("%lf", &num);
if (num <= 0.0) {
if (num == 0.0)
printf("You entered 0.");
else
printf("You entered a negative number.");
}
else
printf("You entered a positive number.");

return 0;
}
```

4. Find the Largest Number Among Three Numbers.
   In C programming, the largest element of a number is the number with the highest numerical value of the three numbers.

   For example, if three numbers are given, 1, 2, 3, the largest of the three numbers is 3.

   Problem Description
   Write a C program that takes the 3 numbers and finds the largest number Aamong three numbers.

   Program :

```
#include <stdio.h>

int main() {

double n1, n2, n3;

printf("Enter three different numbers: ");
scanf("%lf %lf %lf", &n1, &n2, &n3);

if (n1 >= n2 && n1 >= n3)
printf("%.2f is the largest number.", n1);
if (n2 >= n1 && n2 >= n3)
printf("%.2f is the largest number.", n2);

if (n3 >= n1 && n3 >= n2)
printf("%.2f is the largest number.", n3);

return 0;
}
```

5. Swap Two Numbers
   Swapping two numbers in C programming means swapping the values of two variables.
   For example, there are two variables m & n. Value of m is “2” & value of n is “3”.
   Before Swapping: m value = 2; n value = 3
   After Swapping: m value = 3; n value = 2

   Program :

```
#include<stdio.h>
int main() {
double first, second, temp;
printf("Enter first number: ");
scanf("%lf", &first);
printf("Enter second number: ");
scanf("%lf", &second);

temp = first;

first = second;

second = temp;

printf("\nAfter swapping, first number = %.2lf\n", first);
printf("After swapping, second number = %.2lf", second);
return 0;
}
```

6. Find the Number of Integers Divisible by 5
   Problem Description
   1. This problem takes the range as input and finds the number of integers divisible by 5 in the given range.
   2. Also finds the sum of all integers that are divisible by 5 in the given range.

Program :

```
#include <stdio.h>

void main()
{
int i, num1, num2, count = 0, sum = 0;

printf("Enter the value of num1 and num2 \n");
scanf("%d %d", &num1, &num2);
printf("Integers divisible by 5 are \n");
for (i = num1; i < num2; i++)
{
if (i % 5 == 0)
{
printf("%3d,", i);
count++;
sum = sum + i;
}
}
printf("\n Number of integers divisible by 5 between %d and %d =
%d\n", num1, num2, count);
printf("Sum of all integers that are divisible by 5 = %d\n", sum);
}
```

7. Check if Two Numbers are Equal
   Problem Description
   This problem accepts two integers and check if they are equal or not.

   Program :

```
#include <stdio.h>
int areEqual(int num1, int num2) {
return (num1 == num2);
}

int main() {
int num1, num2;
printf("Enter the first integer: ");
scanf("%d", &num1);

printf("Enter the second integer: ");
scanf("%d", &num2);
if (areEqual(num1, num2)) {
printf("The two numbers are equal.\n");
} else {
printf("The two numbers are not equal.\n");
}

return 0;
}
```

8. Sum of Digits
   Sum of digits problem allows a user to enter any number, divide that number into individual numbers, and sum those individual numbers.

   Example 1:
   Given number = 14892 => 1 + 4 + 8 + 9 + 2 = 24.
   Sum of digits of a given number "14892" is 24.

   Example 2:
   Given number = 3721 => 3 + 7 + 2 + 1 = 13.
   Sum of digits of a given number "3721" is 13.

   Program :

```
#include<stdio.h>
 int main()
{
int n,sum=0,m;
printf("Enter a number:");
scanf("%d",&n);
while(n>0)
{
m=n%10;
sum=sum+m;
n=n/10;
}
printf("Sum is=%d",sum);
return 0;
}
```

9. Increment by 1 to all the Digits of a Given Integer
   Problem Description

Increases 1 to all of the given integer digit and print the sum of all digits.

Program :

```
#include <stdio.h>

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

printf("Enter a number: ");
scanf("%d", &number);
while (number)
{
remainder = number % 10;
sum  = sum + (remainder + 1);
number /= 10;
}
printf("increasing 1 to all digits:  %d", sum);
return 0;
}
```

10. Multiplication Table
    A multiplication table of numbers is created by multiplying a constant integer by a number of repetitions ranging from 1 to 10.

Program :

```
#include <stdio.h>

int main() {

int n;

printf("Enter an integer: ");

scanf("%d", &n);


for (int i = 1; i <= 10; ++i) {

printf("%d * %d = %d \n", n, i, n * i);

}

return 0;
}
```

11. Count the Number of Vowels and Consonants in a Sentence

    Problem Description

    This problem takes the sentence as input and counts the number of vowels & consonants in a sentence.

Program :

```
#include <stdio.h>
int main() {
char line[150];
int vowels, consonant, digit, space;
vowels = consonant = digit = space = 0;
printf("Enter a line of string: ");
fgets(line, sizeof(line), stdin);
for (int i = 0; line[i] != '\0'; ++i) {
line[i] = tolower(line[i])
if (line[i] == 'a' || line[i] == 'e' || line[i] == 'i' ||
line[i] == 'o' || line[i] == 'u') {
++vowels;
}
else if ((line[i] >= 'a' && line[i] <= 'z')) {
++consonant;
}
else if (line[i] >= '0' && line[i] <= '9') {
++digit;
}
else if (line[i] == ' ') {
++space;
}
}
printf("Vowels: %d", vowels);
printf("\nConsonants: %d", consonant);
printf("\nDigits: %d", digit);
printf("\nWhite spaces: %d", space);
return 0;
}
```

12. Accept the Height of a Person & Categorize as Taller, Dwarf & Average
    Problem Description
    This problem accepts the height of a person as input and categorizes as taller, dwarf & average.

    Program :

```
#include <stdio.h>
void main()
{
float height;

printf("Enter  the Height (in centimetres) \n");
scanf("%f", &height);
if (height < 150.0)
printf("Dwarf \n");
else if ((height >= 150.0) && (height <= 165.0))
printf(" Average Height \n");
else if ((height > 165.0) && (height <= 195.0))
printf("Taller \n");
else
printf("Abnormal height \n");
}
```

13. Prime Number
    A prime number is a natural number that is greater than 1 and is only divisible by 1 and itself. In other words, no number except the number itself, and 1 can divide a prime number.
    Example: 2, 3, 5, 7, 11, 13, 17, 19 …., etc.
    Problem Description

check if a given number is Prime number. If the number is Prime, then display it is a prime number else display it is not a prime number.

Program :

```
int main() {

int n, i, flag = 0;
printf("Enter a positive integer: ");
scanf("%d", &n);

if (n == 0 || n == 1)
flag = 1;

for (i = 2; i <= n / 2; ++i) {

if (n % i == 0) {
flag = 1;
break;
}
}

if (flag == 0)
printf("%d is a prime number.", n);
else
printf("%d is not a prime number.", n);

return 0;
}
```

14. Check Whether a Given Number is Perfect Number
A perfect number is a number that is equal to the sum of its proper divisors. For example, the divisors of 6 are 1, 2 and 3. The sum of the proper divisors of 6 is 1 + 2 + 3 = 6, which is a perfect number. The sum of the proper divisors of 28 is 1 + 2 + 4 + 7 + 14 = 28, which is also a perfect number.

Problem Description
Ask the user for a number and then check whether the number is a perfect number or not.

Program :

```c
#include <stdio.h>

int main()
{
int i, num, sum = 0;

printf("Enter any number to check perfect number: ");
scanf("%d", &num);

for(i = 1; i <= num / 2; i++)
{

if(num%i == 0)
{
sum += i;
}
}

if(sum == num && num > 0)
{
printf("%d is PERFECT NUMBER", num);
}
else
{
printf("%d is NOT PERFECT NUMBER", num);
}

return 0;
}
```

15. Check Armstrong Number
    Armstrong Number in C: An Armstrong number is an n-digit base b number such that the sum of its (base b) digits raised to the power n is the number itself. Armstrong numbers are 0, 1, 153, 370, 371, 407, etc.

    Armstrong Number Formula: wxyz = pow(w,n) + pow(x,n) + pow(y,n) + pow(z,n)

Program :

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

```
int main() {
int num, originalNum, remainder, result = 0;
printf("Enter a three-digit integer: ");
scanf("%d", &num);
originalNum = num;

while (originalNum != 0) {

remainder = originalNum % 10;

result += remainder * remainder * remainder;

originalNum /= 10;
}

if (result == num)
printf("%d is an Armstrong number.", num);
else
printf("%d is not an Armstrong number.", num);

return 0;
}
```

16. Reverse a Number
    Reverse a Number means moving the digit at the last position to the first position and vice versa.

    For example, if the given number is “1234”, the reverse number will be “4321”.

Program :

```
#include <stdio.h>

int main() {

int n, reverse = 0, remainder;

printf("Enter an integer: ");
scanf("%d", &n);

while (n != 0) {
remainder = n % 10;
reverse = reverse * 10 + remainder;
n /= 10;
}

printf("Reversed number = %d", reverse);

return 0;
}
```

17. Reverse a Number and Check if it is a Palindrome
    Problem Description
    Accepts an integer, reverse it and also checks if it is a palindrome or not.

    Program :

```
#include <stdio.h>

int main()
{
int num, temp, remainder, reverse = 0;

printf("Enter an integer \n");
scanf("%d", &num);

temp = num;
while (num > 0)
{
remainder = num % 10;
reverse = reverse * 10 + remainder;
num /= 10;
}
printf("Given number is = %d\n", temp);
printf("Its reverse is  = %d\n", reverse);
if (temp == reverse)
printf("Number is a palindrome \n");
else
printf("Number is not a palindrome \n");
}
```

18. C Program to Add Two Binary Numbers
    Problem Description
    This program finds the sum of two binary numbers.

    Program :

```
#include <stdio.h>

int main()
{

long binary1, binary2;
int i = 0, remainder = 0, sum[20];

printf("Enter the first binary number: ");
scanf("%ld", &binary1);
printf("Enter the second binary number: ");
scanf("%ld", &binary2);
while (binary1 != 0 || binary2 != 0)
{
sum[i++] =(binary1 % 10 + binary2 % 10 + remainder) % 2;
remainder =(binary1 % 10 + binary2 % 10 + remainder) / 2;
binary1 = binary1 / 10;
binary2 = binary2 / 10;
}
if (remainder != 0)
sum[i++] = remainder;
--i;
printf("Sum of two binary numbers: ");
while (i >= 0)
printf("%d", sum[i--]);
return 0;
```

}

19. Find Prime Numbers in a Given Range
    Problem Description
    Take the range and finds all the prime numbers between the range and also prints the number of prime numbers.

```
Case:1
Enter the value of num1 and num2
70 85
Prime numbers are
71
73
79
83
```

Program :

```
#include <stdio.h>


int main() {
int min, max, i, j, count = 0;
printf("Enter Your First Number\n");
scanf("%d", &min);
printf("Enter Your Last Number\n");
scanf("%d", &max);
for(i=min; i<=max; i++) {
for(j=1; j<=i; j++) {
if(i % j == 0) {
count++;
}
}
if(count==2) {
printf("%d\t",i);
}
}
return 0;
}
```

20. Leap Year
    Leap Year: A year is a Leap Year if it satisfies the following conditions:

    The year is exactly divisible by 400 (such as 2000,2400) or, The year is exactly divisible by 4 (such as 2008, 2012, 2016) and not a multiple of 100 (such as 1900, 2100, 2200).

Program :

```
#include <stdio.h>
int main() {
int year;
printf("Enter a year: ");
scanf("%d", &year);
if (year % 400 == 0) {
printf("%d is a leap year.", year);
}
else if (year % 100 == 0) {
printf("%d is not a leap year.", year);
else if (year % 4 == 0) {
printf("%d is a leap year.", year);
}
else {
printf("%d is not a leap year.", year);
}
return 0;
}
```

21. Fibonacci Series

Fibonacci series are the numbers in the sequence 0, 1, 1, 2, 3, 5, 8, 13, 21….. The series in the Fibonacci sequence is equal to the sum of the previous two terms. The Fibonacci sequence's first two terms are 0 and 1 respectively.

Program :

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

int i, n;

int t1 = 0, t2 = 1;

int nextTerm = t1 + t2;

printf("Enter the number of terms: ");
scanf("%d", &n);
printf("Fibonacci Series: %d, %d, ", t1, t2);

for (i = 3; i <= n; ++i) {
printf("%d, ", nextTerm);
t1 = t2;
t2 = nextTerm;
nextTerm = t1 + t2;
}

return 0;
}
```

22. Factorial

When you multiply a positive integer by all the integers smaller than that positive integer, you get its factorial.
For example, factorial of 3 is 3! = 1*2*3 = 6 and factorial of 6 is 6! = 6 * 5 * 4 * 3 * 2 * 1 which equals to 720.
By default, the factorial of 0 is 1, and Factorial of a negative number is not defined.

Program :

```
#include <stdio.h>
int main() {
int n, i;
unsigned long long fact = 1;
printf("Enter an integer: ");
scanf("%d", &n);


if (n < 0)
printf("Error! Factorial of a negative number doesn't exist.");
else {
for (i = 1; i <= n; ++i) {
fact *= i;
}
printf("Factorial of %d = %llu", n, fact);
}

return 0;
}
```

23. Floyd’s Triangle

Floyds Triangle in C is a right-angled triangular array of natural numbers. It is defined by filling the rows of the triangle with consecutive numbers, starting with a 1 in the top left corner: 1. Number of rows of Floyd’s triangle to print is entered by the user. For loop is used to print the output of the program.

Example:

A Floyd’s triangle is a triangle in which each number is the sum of the two numbers above it. For example, the first row of the Floyd’s triangle is 1, the second row is 2 + 1 = 3, and so on. The following is a diagram of the Floyd’s triangle:

```
1
2 3
4 5 6
7 8 9 10
```

Program :

```
#include <stdio.h>

int main()
{
int num, i, j, k = 1;

printf( " Enter a number to define the rows in Floyd's triangle: \n");
scanf( "%d", &num);
// use nested for loop
// outer for loop define the rows and check rows condition
for (i = 1; i <= num; i++)
{
// inner loop check j should be less than equal to 1 and print the data.
for (j = 1; j <= i; j++)
{
printf(" %2d", k++); // print the number
}
printf( "\n");
}
return 0;
}
```

24. Pascal Triangle

Pascal Triangle is a pattern similar to a triangle. Firstly, 1 is placed at the top, and then we start putting the numbers in a triangular pattern. The numbers which we get in each step are the addition of the above two numbers.

```
Enter the Number of Rows in the Pascal Triangle:: 6
          1
        1    1
      1    2    1
    1    3    3    1
  1    4    6    4    1
1    5    10   10   5    1
```

Program :

```
#include<stdio.h>

long factorial(int);

int main()
{
int i, n, c;

printf("How many rows you want to show in pascal triangle?\n");

scanf("%d",&n);

for ( i = 0 ; i < n ; i++ )
{
for ( c = 0 ; c <= ( n - i - 2 ) ; c++ )
printf(" ");
for( c = 0 ; c <= i ; c++ )
printf("%ld ",factorial(i)/(factorial(c)*factorial(i-c)));
printf("\n");
}
return 0;
}

long factorial(int n)
{
int c;
long result = 1;

for( c = 1 ; c <= n ; c++ )
result = result*c;
return ( result );
}
```

25. A star pattern is a pattern that shows up as a staircase of stars.

```
*
**
***
****
*****
```

program :

```
#include <stdio.h>

int main(void)
{
int i, j, n;
printf("Enter the number of rows: ");
scanf("%d", &n);
for (i = 1; i <= n; i++)
{
for (j = 1; j <= i; j++)
{
printf("*");
}
printf("\n");
}
return 0;
}
```

26. Rhombus Star Pattern in C

```
*
***
*****
***
*
```

Program :

```
#include<stdio.h>
int main()
{
int i, j, k, rows;
printf("Enter Rhombus Star Pattern Rows =  ");
scanf("%d", &rows);

printf("Rhombus Star Pattern\n");
for(i = rows; i >= 1; i--)
{
for(j = 1; j <= i - 1; j++)
{
printf(" ");
}
for(k = 1; k <= rows; k++)
{
printf("*");
}
printf("\n");
}
return 0;
}
```

27. Diamond Star Pattern

```
Enter the number of rows: 5
    *
   ***
  *****
   ***
    *
```

Program :

```
#include <stdio.h>

int main()

{

int i,j,n;

char ch;


printf("Enter number of rows: ");

scanf("%d%c",&n,&ch);

printf("Enter the symbol: ");

ch=getchar();

for(i=1;i<=n;i++)

{
```

```
for(j=1;j<=n-i;j++)
{
printf(" ");
}

for(j=1;j<=i*2-1;j++)


{
printf("%c",ch);
}
printf("\n");


}


for(i=n-1;i>0;i--)
{
for(j=1;j<=n-i;j++)
{
printf(" ");
}
for(j=1;j<=i*2-1;j++)
{
printf("%c",ch);
}
printf("\n");
}
```

```
return 0;

}
```

28. Find the Area of a Circle
    Area of circle is defined as pi*r*r where pi is a constant whose value is (22/7 or 3.142) and r is the radius of a circle.

    Formula to calculate the area of circle is: Area = pi*r*r

```
Enter Radius of Circle:
10
The area of Circle with radius 10 is: 314.16
```

Program :

```
#include <stdio.h>
int main()
{
float radius, area;

printf("Enter the radius of a circle\n");

scanf("%f", &radius);

area = 3.14159*radius*radius;

printf("Area of the circle = %.2f\n", area);

return 0;
}
```

29. Find the Area of a Triangle
    The area of a triangle is defined as the total area bounded by the three sides of a given triangle.

    Area of a Triangle Formula:

    If the base and height are given, the area of the triangle is determined using the formula A = 1/2∗b∗h

```
Enter Base and Height: 10 5
Area of Triangle is 25.00
```

Program :

```
#include <stdio.h>

int main()
{
float base, height, area;

printf("Enter base of the triangle: ");
scanf("%f", &base);
printf("Enter height of the triangle: ");
scanf("%f", &height);

area = (base * height) / 2;


printf("Area of the triangle = %.2f sq. units", area);

return 0;
}
```

30. Find GCD and LCM of Two Integers
    GCD (Greatest Common Divisor)
    GCD stands for Greatest Common Divisor. GCD of two numbers is the largest positive integer that completely divides both the given numbers.

    Example: GCD(10,15) = 15, GCD(12,15) = 3.

    LCM (Least Common Multiple)
    LCM stands for Least Common Multiple. It is a method to find the lowest common multiple between the two numbers. LCM of two numbers is the lowest possible number that is divisible by both numbers.

    Examples: LCM(10,15) = 30, LCM(12,15) = 60.

```
Enter two numbers:
12 15
GCD of 12 and 15 = 3
LCM of 12 and 15 = 60
```

Program :

```
#include <stdio.h>

int main()
{
int num1, num2, gcd, lcm, remainder, numerator, denominator;

printf("Enter two numbers:\n");
scanf("%d %d", &num1, &num2);

numerator = (num1>num2)?num1:num2;
denominator = (num1<num2)?num1:num2;
remainder = numerator % denominator;

while (remainder != 0)
{
numerator   = denominator;
denominator = remainder;
remainder   = numerator % denominator;
}
gcd = denominator;
lcm = num1 * num2 / gcd;
printf("GCD of %d and %d = %d\n", num1, num2, gcd);
printf("LCM of %d and %d = %d\n", num1, num2, lcm);
return 0;
}
```

31. Find HCF of Two Numbers
    HCF stands for Highest Common Factor. HCF of two numbers is the largest positive integer that completely divides both the given numbers.

    Example: HCF(10,15) = 15, HCF(12,15) = 3.

```
Enter two numbers
12 15
HCF of 12 and 15 = 3.
```

Program :

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

```
int n1, n2, i, gcd;

printf("Enter two integers: ");
scanf("%d %d", &n1, &n2);

for(i=1; i <= n1 && i <= n2; ++i)
{
if(n1%i==0 && n2%i==0)
gcd = i;
}

printf("G.C.D of %d and %d is %d", n1, n2, gcd);

return 0;
}
```

32. Compare Two Strings

String is a sequence of characters terminated by the special character '\0'. Strings can be compared with or without using the string function.

Example:

String1="Hello"	String2="Hello"	Both string are equal
String1="Hello"	String2="Hell"	String1 is greater
String1="Hello"	String2="Helz"	String2 is greater

```
Enter the First string
hello
Enter the Second string
hell
First string is greater than Second string
```

Program :

```
#include <stdio.h>
#include<string.h>
int main()
{
char str1[20];
char str2[20];
int value;

printf("Enter the first string : ");
scanf("%s",str1);

printf("Enter the second string : ");

scanf("%s",str2);

value=strcmp(str1,str2);

if(value==0)

printf("strings are same");

else

printf("strings are not same");

return 0;
}
```

33. Check Whether a Number is Palindrome or Not
    A number is said to be a palindrome number if it reads the same forward and backward i.e., on reversing the digits of the number we get the same number.

```
Enter the number: 121
121 is a palindrome number.
```

```
Enter the number: 342
342 is not a palindrome number.
```

Program :

```
#include <stdio.h>
int main()
{
int n, reversed = 0, remainder, original;

printf("Enter an integer: ");

scanf("%d", &n);

original = n;
while (n != 0) {
remainder = n % 10;

reversed = reversed * 10 + remainder;

n /= 10;
}

if (original == reversed)

printf("%d is a palindrome.", original);

else

printf("%d is not a palindrome.", original);

return 0;
}
```

34. String Palindrome
String Palindrome: A palindrome is a word, phrase or sentence that reads the same backward or forward. A string is said to be a palindromic string when we traverse it from start to end or end to start then we get the same result.

```
Enter a string: sanfoundry
sanfoundry is not a palindrome

Enter a string: malayalam
malayalam is a palindrome
```

program :

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

int main() {
char string1[20];
int i, length;
int flag = 0;

printf("Enter a string: ");
scanf("%s", string1);
length = strlen(string1);

for (i = 0; i < length / 2; i++) {
if (string1[i] != string1[length - i - 1]) {
flag = 1;
break;
}
}

if (flag) {
printf("%s is not a palindrome\n", string1);
} else {
printf("%s is a palindrome\n", string1);
}

return 0;
}
```

35. Anagram
    Anagram: Two strings are said to be anagrams if they satisfy two conditions, the length of both strings must be equal to each other and second the strings must have the same set of characters.
    Example 1:

    First String = “hectare” and Second String = “teacher”

    Case 1:
    Lengths must be equal to each other.
    length of “hectare” = 7
    length of “teacher” = 7
    Case 1 passed.

    Case 2:
    Set of characters in
    hectare {‘h’ , ’e’ , ’c’ , ’t’ , ’a’ , ’r’ , ’e’}
    teacher {‘t’ , ’e’ , ’a’ , ’c’ , ’h’ , ’e’, ’r’}

    Every character from the first string has a similar character to it in the other string.
    Case 2 passed.

    ”teacher” and ”hectare” are anagrams.

```
Enter the string
study
Enter another string
dusty
"study" and "dusty" are anagrams.
```

Program :

```
#include <stdio.h>
int check_anagram(char [], char []);

int main()
{
char a[1000], b[1000];

printf("Enter two strings\n");
gets(a);
gets(b);

if (check_anagram(a, b))
printf("The strings are anagrams.\n");
else
printf("The strings aren't anagrams.\n");

return 0;
}

int check_anagram(char a[], char b[])
{
int first[26] = {0}, second[26] = {0}, c=0;

while (a[c] != '\0') {
first[a[c]-'a']++;
c++;
}

c = 0;

while (b[c] != '\0') {
second[b[c]-'a']++;
c++;
}

for (c = 0; c < 26; c++)
if (first[c] != second[c])
return 0;

return 1;
}
```

36. Calculate the Power of a Number

    For example: In the case of 2 3
    2 is the base number
    3 is the exponent
    And, the power is equal to 2*2*2

    Sample input
    Base number: 2
    Exponent number: 3
    Output:
    8

    program :

```
#include <stdio.h>
int main() {
   int base, exp;
   long double result = 1.0;
   printf("Enter a base number: ");
   scanf("%d", &base);
   printf("Enter an exponent: ");
   scanf("%d", &exp);

   while (exp != 0) {
      result *= base;
      --exp;
   }
   printf("Answer = %.0Lf", result);
   return 0;
}
```

37. Print the sum of all even numbers between 1 and 100.
    Example Solution: Sum of even numbers between 1 and 100: 2550

Program :

```
#include <stdio.h>
int main(){
for (int i = 2; i <= 100; i++)
{
if (i % 2 == 0)
{
printf("%d\n", i);
}
}
return 0;
}
```

38. Check if a given number is a perfect square or not.
    Sample Input: Enter a number: 25
    Sample Output: 25 is a perfect square.

    Program :

```
#include <stdio.h>
#include<math.h>
int main()
{
int num;
int iVar;
float fVar;

printf("Enter an integer number: ");
scanf("%d",&num);

fVar=sqrt((double)num);
iVar=fVar;

if(iVar==fVar)
printf("%d is a perfect square.",num);
else
printf("%d is not a perfect square.",num);

return 0;
}
```

39. Find the sum of all even digits in a given number.
Sample:
Input: Enter a number: 356824
Output: Sum of even digits: 20

Program :

```
#include<stdio.h>
int sum_of_even_digits(int n) {
int r, sum = 0;

while (n > 0) {
r = n % 10;
n = n / 10;

if (r % 2 == 0){
sum = sum + r;
}
}
return sum;
}
int main() {
int n;
printf("Enter a Number: ");
scanf("%d", &n);
printf("Sum of Even Digits: %d", sum_of_even_digits(n));
}
```

40. Swap two numbers without using a temporary variable.
Sample:
Input:
Enter first number: 10
Enter second number: 20
Output:
Before swapping: num1 = 10, num2 = 20
After swapping: num1 = 20, num2 = 10

Program :

```
#include <stdio.h>
int main()
{
int x = 10, y = 20;


x = x + y;
y = x - y;
x = x - y;

printf("After Swapping: x = %d, y = %d", x, y);

return 0;
}
```

41. Find the Number of Elements in an Array

```
array[] = {15, 50, 34, 20, 10, 79, 100};
Size of the given array is 7
```

Program :

```
#include <stdio.h>

int main()
{
double arr[] = {15, 50, 34, 20, 10, 79, 100};

int n;

n = sizeof(arr) / sizeof(arr[0]);

printf("Size of the array is: %d\n", n);

return 0;
}
```

42. Delete an Element from an Array

Example: arr[6] = {12,65,32,75,48,11}

```
Value:  12  65  32  75  48  11
         ↑   ↑   ↑   ↑   ↑   ↑
Index:   0   1   2   3   4   5
```

The Element we are deleting here is “75”.

Original Array:

```
12  65       32       75       48       11
```

New Array:

```
12  65       32       48       11
```

Program :

```
#include <stdio.h>

int findElement(int arr[], int n, int key);

int deleteElement(int arr[], int n, int key)
{

int pos = findElement(arr, n, key);

if (pos == -1) {
printf("Element not found");
return n;
}

int i;
for (i = pos; i < n - 1; i++)
arr[i] = arr[i + 1];

return n - 1;
}

int findElement(int arr[], int n, int key)
{
int i;
for (i = 0; i < n; i++)
if (arr[i] == key)
return i;

return -1;
}
int main()
{
```

```
int i;
int arr[] = { 10, 50, 30, 40, 20 };

int n = sizeof(arr) / sizeof(arr[0]);
int key = 30;

printf("Array before deletion\n");
for (i = 0; i < n; i++)
printf("%d ", arr[i]);

n = deleteElement(arr, n, key);

printf("\nArray after deletion\n");
for (i = 0; i < n; i++)
printf("%d ", arr[i]);

return 0;
}
```

43. Find Sum of Array Elements using Pointer
    Expected Input and Output
    If we are entering 5 elements (N = 5), with array element values as 4, 9, 10, 56 and 100 then,
    Sum of Elements of the array will be: 4 + 9 + 10 + 56 + 100 = 179

    Program :

```
#include <stdio.h>

int main() {

int arr[100], size;
int *ptr, sum = 0;

printf("Enter the size of the array: ");
scanf("%d", &size);

printf("Enter array elements: ");
for (int i = 0; i < size; i++) {
scanf("%d", &arr[i]);
}
ptr = arr;

for (int i = 0; i < size; i++) {
sum = sum + *ptr;
ptr++;
}

printf("The sum of array elements is: %d", sum);

return 0;

}
```

44. Print all Non Repeated Elements in an Array

```
Enter size of the array: 6

Enter 6 elements of an array: 12
10
4
10
12
56

The array after removing duplicates is:  12 10 4 56
```

Program :

```
#include<stdio.h>
int nonRepeatingint(int a[],int n)
{
int c,i,j,count=0;
for(i = 0; i < n; i++)
{
c=0;
for(j = 0; j < n; j++)
{
if(a[i]==a[j] && i!=j)
{
c++;
break;
}
}
if(c==0)
{
printf(" %d",a[i]);
count=1;
}
}
return count;
}
int main()
{
int n,i,j,c,count=0;
printf("\nEnter the number of elements in an array: ");
scanf("%d",&n);
int a[n];
printf("\nEnter %d elements in an array :\n",n);
for(i = 0; i < n; i++)
{
scanf("%d",&a[i]);
}
printf("\nNon repeating elements in an array :");
count=nonRepeatingint(a,n);
if(count==0)
printf("\nThere are no Non repeating elements in an array ");
return 0;
}
```

45. Cyclically Permute the Elements of an Array

```
Enter the value of the n = 4
Enter the numbers
3
40
100
68
Cyclically permuted numbers are given below
40
100
68
3
```

Program :

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

```
int main ()
{

int i, n, number[30];
printf("Enter the value of the n = ");
scanf("%d", &n);

printf("Enter the numbers\n");
for (i = 0; i < n; ++i)
{
scanf("%d", &number[i]);
}

number[n] = number[0];
for (i = 0; i < n; ++i)
{
number[i] = number[i + 1];
}

printf("Cyclically permuted numbers are given below \n");
for (i = 0; i < n; ++i)
printf("%d\n", number[i]);
return 0;
}
```

46. Find Missing Numbers in Array

```
Enter size of array : 6
Enter elements into array :
1
2
3
5
6
Missing element is : 4
```

Program :

```
#include <stdio.h>

int main()
{
int n, i, j, c, t, b;
printf("Enter size of array : ");

scanf("%d", &n);

int array[n - 1];

printf("Enter elements into array : \n");

for (i = 0; i < n - 1; i++)

scanf("%d", &array[i]);

b = array[0];

for (i = 1; i < n - 1; i++)

b = b ^ array[i];

for (i = 2, c = 1; i <= n; i++)

c = c ^ i;

c = c ^ b;

printf("Missing element is : %d \n", c);

return 0;
}
```

47. Find Union and Intersection of Two Arrays

```
Enter the elements of Array 1:

Enter element 1: 12

Enter element 2: 34

Enter element 3: 23

Enter element 4: 56

Enter element 5: 45
```

```
Elements of Array 1: { 12 34 23 56 45 }

Sorted elements of Array 1: { 12 23 34 45 56 }

Enter the elements of Array 2:

Enter element 1: 34

Enter element 2: 56

Enter element 3: 12

Enter element 4: 78

Enter element 5: 66


Elements of Array 2: { 34 56 12 78 66 }

Sorted elements of Array 2: { 12 34 56 66 78 }

Intersection is: { 12 34 56 }

Union is: { 12 23 34 45 56 66 78 }
```

Program :

```
#include <stdio.h>
void findUnion(int arr1[], int m, int arr2[], int n) {
int i = 0, j = 0;

while (i < m && j < n) {
if (arr1[i] < arr2[j]) {
printf("%d ", arr1[i]);
i++;
} else if (arr2[j] < arr1[i]) {
printf("%d ", arr2[j]);
j++;
} else {
printf("%d ", arr1[i]);
i++;
j++;
}
}
while (i < m) {
printf("%d ", arr1[i]);
i++;
}
while (j < n) {
printf("%d ", arr2[j]);
j++;
}
}
void findIntersection(int arr1[], int m, int arr2[], int n) {
int i = 0, j = 0;

while (i < m && j < n) {
if (arr1[i] < arr2[j]) {
i++;
} else if (arr2[j] < arr1[i]) {
j++;
} else {
printf("%d ", arr1[i]);
i++;
j++;
}
}
}

int main() {
int arr1[] = {1, 2, 3, 4, 5};
int m = sizeof(arr1) / sizeof(arr1[0]);

int arr2[] = {3, 4, 5, 6, 7};
int n = sizeof(arr2) / sizeof(arr2[0]);

printf("Union: ");
findUnion(arr1, m, arr2, n);

printf("\nIntersection: ");
findIntersection(arr1, m, arr2, n);
```

```
return 0;
}
```

48. Split the Array and Add First Part to the End

```
Enter the value of n
4
enter the numbers
3
678
345
876
Enter the position of the element to split the array
3
The resultant array is
876
3
678
345
```

Program :

```
#include <stdio.h>

#define MAX_SIZE 100

int main() {
int arr[MAX_SIZE], size, splitIndex;

printf("Enter size of the array: ");
scanf("%d", &size);

printf("Enter elements of the array:\n");
for (int i = 0; i < size; ++i)
scanf("%d", &arr[i]);

printf("Enter the index to split the array: ");
scanf("%d", &splitIndex);

if (splitIndex >= 0 && splitIndex < size) {
for (int i = 0; i < splitIndex; ++i)
printf("%d ", arr[(i + splitIndex) % size]);
printf("\n");
} else {
printf("Invalid split index.\n");
}

return 0;
}
```

49. Matrix Multiplication

Example 1:

[1 4
 3 2] *
[1 2
 2 1]
= [1∗1+3∗24∗1+2∗21∗2+3∗14∗2+2∗1] = [78510]

Program :

```
#include <stdio.h>

int main() {
int a[2][2], b[2][2], result[2][2];
printf("Enter elements of matrix A:\n");
for (int i = 0; i < 2; ++i)
for (int j = 0; j < 2; ++j)
scanf("%d", &a[i][j]);

printf("Enter elements of matrix B:\n");
for (int i = 0; i < 2; ++i)
for (int j = 0; j < 2; ++j)
scanf("%d", &b[i][j]);

for (int i = 0; i < 2; ++i)
for (int j = 0; j < 2; ++j)
for (int k = 0; k < 2; ++k)
result[i][j] += a[i][k] * b[k][j];

printf("Resultant matrix:\n");
for (int i = 0; i < 2; ++i) {
for (int j = 0; j < 2; ++j)
printf("%d ", result[i][j]);
printf("\n");
}

return 0;
}
```

50. Find the Perimeter of a Circle, Rectangle and Triangle
perimeter of rectangle: 2 * (a + b)
perimeter of General triangle: a + b + c
perimeter of Equilateral triangle: 3 * a
perimeter of Right angled triangle: width + height + sqrt(width ^ 2 + height ^ 2)
perimeter of circle: 2 * pi * r

Program :

```
#include <stdio.h>


#define PI 3.14159

int main() {

double radius;
printf("Enter the radius of the circle: ");
scanf("%lf", &radius);
double circlePerimeter = 2 * PI * radius;

double length, width;
printf("Enter the length of the rectangle: ");
scanf("%lf", &length);
printf("Enter the width of the rectangle: ");
scanf("%lf", &width);
double rectanglePerimeter = 2 * (length + width);

double side1, side2, side3;
printf("Enter the lengths of the sides of the triangle:\n");sss
scanf("%lf %lf %lf", &side1, &side2, &side3);
double trianglePerimeter = side1 + side2 + side3;
printf("Perimeter of the Circle: %.2lf\n", circlePerimeter);
printf("Perimeter of the Rectangle: %.2lf\n", rectanglePerimeter);
printf("Perimeter of the Triangle: %.2lf\n", trianglePerimeter);

return 0;
}
```