1.
```c
#include <stdio.h>
int main() {
/* printf() displays the string
 inside quotation */
  printf ("Hello , World!");
  return 0;
}
```

Hello, World!

#Hello World

★ Hello world

2.
```c
#include <stdio.h>
int main()
{
  int number;
  printf ("Enter an integer:");
  // reads and stores input
  scanf ("%d", &number);
  // displays output
  printf ("You enterered : %d", number);
  return 0;
  }
```

Enter an integer : 6
You entered : 6

#Print an integer (Entered by the user)

★ Print an integer

3.
```c
#include <stdio.h>
int main(){
  int number1, number2, sum;
  printf ("Enter two integers:");
  scanf("%d %d", &number1, &number2);
  //calculating sum
  sum = number1 + number2;
  printf ("%d + %d = %d", number1, number2, sum);
  return 0;
  }
```

#Add two integers

★ Add two integers

Enter two integers: 6 7
6 + 7 = 13

4.
```
include <stdio.h>
int main() {
  float a, b, product;
  printf("Enter two numbers:");
  scanf("%f %f", &a, &b);
  // calculating product
  product = a * b
  // Result up to 2 decimal point is displayed
     using %.2f
  printf("product = %.2f", product);
  return 0;
}
```

multiply two floating point numbers

Enter two numbers: 6 7
Product = 42.00

*** Multiply two floating point numbers**

5.
```
# include <stdio.h>
int main(){
  char c;
  printf("Enter a character:");
  scanf("%c", &c);
  // %d displays the integer value of a character
  // %c displays the actual character
  printf("ASCII value of %c = %d", c, c);
  return 0;
}
```

Find ASCII values of character

Enter a character : j
ASCII value of j = 106

*** Find ASCII value of character**

6.
```
#include <stdio.h>
int main (){
  int dividend, divisor, quotient, remainder;
  printf ( "Enter dividend :");
  Scanf ( " %d ", & dividend );
  printf (" Enter  divisor :");
  scanf (" %d ", & divisor);
  // computes quotient
quotient = dividend /divisor;
  // computes remainder
remainder = dividend % divisor;
  printf ( " Quotient = %d \n", quotient);
  printf ("Remainder = %d ", remainder);
  return 0;
}
```

Compute Quotient and Remainder

Compute Quotient and Remainder.

```
Enter dividend : 6
Enter divisor : 6
Quotient = 1
Remainder = 0.
```

7.
```
#include <stdio.h>
int main ()
{
  int intType;
  float floatType;
  double doubleType;
  char charType;
```

Find the size of int, float, double and char

```
//sizeof evaluates the size of a variable
printf("Size of int : %ld bytes \n", sizeof(intType));
printf("Size of float : %ld bytes \n",
                    sizeof(floatType));
printf("Size of double : %ld bytes \n",
          sizeof(doubleType));
printf("Size of char : %ld bytes \n", sizeof(charType));
return 0;
}
```

Find the size of int, float, double and char

```
Size of int : 4 bytes
Size of float : 4 bytes
Size of double : 8 bytes
Size of char : 1 byte
```

8.
```
# include <stdio.h>
int main()
{
  int a;
  long b;        //equivalent to long int b;
  long long c;   //equivalent to long long int c;
  double e;
  long double f;
  printf("size of int = %zu bytes \n", sizeof(a));
  printf("size of long int = %zu bytes \n",
          sizeof(b));
```

Demonstrate the working of long keywords

```
printf("size of long long int = %zu bytes \n",
        sizeof(c));
printf("Size of double = %zu bytes \n", sizeof(e));
printf("Size of long double = %zu bytes \n",
        sizeof(f));

return 0;
}
```

* Demonstrates the working of long keywords

```
Size of int = 4 bytes
Size of long int = 8 bytes
Size of long long int = 8 bytes
Size of double = 8 bytes
Size of long double = 16 bytes
```

9.
```
# include <stdio.h>
int main()
{
double first, second, temp;
printf("Enter first number:");
scanf("%lf", &first);
printf("Enter second number:");
scanf("%lf", &second);
// value of first is assigned to temp
temp = first;
// value of second is assigned to first
first = second;
// value of temp (initial value of first) is
assigned to second.
```

swap two numbers

* Swap two Numbers

```
second = temp;
printf("\n After swapping, first number = %.2lf \n",
        first);
printf("After swapping second number = %.2lf",
        second);

  return 0;
}
```

```
Enter first number: 6
Enter second number: 7

After swapping, first number = 7.00
After swapping, second number = 6.00
```

10. # Check whether a no. is Even or Odd

~~#~~ Check whether a no. is even or odd

```
#include <stdio.h>
int main(){
    int num;
    printf("Enter an integer :");
    scanf("%d", &num);
// True if num is perfectly divisible by 2
    if (num %2 ==0)
        printf("%d is even.", num);
    else
        printf("%d is odd.", num);

    return 0;
}
```

```
Enter an integer :6
6 is even.
```

11. #include <stdio.h>
int main()
{
char c;
int lowercase_vowel, uppercase_vowel;
printf ("Enter an alphabet:");
scanf ("%c", &c);

// evaluates to 1 if variable c is a lowercase vowel
lowercase_vowel = (c == 'a' || c == 'e' || c == 'i' ||
c == 'o' || c == 'u');

//evaluates to 1 if variable c is a uppercase vowel.
uppercase_vowel = (c == 'A' || c == 'E' || c == 'I' ||
c == 'O' || c == 'U');

// evaluates to 1 (true) if c is a vowel
if (lowercase_vowel || uppercase_vowel)
printf("%c is a vowel.", c);
else
printf("%c is a consonant.", c);

return 0;
}

// check whether a character is a vowel or consonant.

Enter an alphabet: y
y is a consonant.

12.
```c
#include <stdio.h>
int main()
{
    double n1, n2, n3;
    printf("Enter three different numbers : ");
    scanf("%lf %lf %lf", &n1, &n2, &n3);
    // if n1 is greater than n2 & n3
    // n1 is the largest
    if (n1>=n2 && n1>=n3)
        printf("Largest number = %.2f", n1);
    // if n2 is greater than n1 & n3
    // n2 is the largest.
    if (n2>=n1 && n2>=n3)
        printf("Largest Number = %.2f", n2);
    // if n3 is greater than n1 and n2
    // n3 is the largest
    if (n3>=n1 && n3>=n2)
        printf("Largest number = %.2f", n3);
    return 0;
}
```

Program to find the Largest no.

Program to find the largest no.

Enter three different numbers : 6 7 8

Largest Number = 8.00

13.
```
# include<math.h>
# include<stdio.h>
int main()
{
 double a, b, c, discriminant, root1, root2,
   realpart, imagpart;
 printf("Enter coefficients a, b and c:");
 scanf("%lf %lf %lf", &a, &b, &c);

 discriminant = b*b - 4*a*c;
 // condition for real and different roots
 if (discriminant > 0){
   root1 = (-b + sqrt(discriminant)) / (2*a);
   root2 =  (-b - sqrt(discriminant)) / (2*a);
   printf("root1 = %.2lf and root2 = %.2lf",
          root1, root2);
 }
 // condition for real and equal roots.
 else if  (discriminant == 0)
 {
   root1= root2 = -b/(2*a);
   printf("root1 = root2 = %.2lf;", root1);
 }

 // if roots are not real.
 else
 {
```

Find the roots of a Quadratic Equation

~~#~~ Find the roots of a Quadratic Equation

```
realpart = -b/(2*a);
imagpart = sqrt(-discriminant)/(2*a);
printf("root1 = %.2lf + %.2lfi and
    root2 = %.2f - %.2fi", realpart,
imgpart, realpart, imagpart);
}
return 0;
}
```

Enter coefficients a, b & c: 6 7 8

root1 = -0.58+1.00i and root2 = -0.58-1.00i

14.
```
#include <stdio.h>
int main()
{
 int year;
printf("Enter year :");
scanf("%d", &year);
if (year%4==0)
{
 // nested if else.
  if (year%100==0)
  {
   if (year%400==0)
     printf("%d is a leap year", year);
```

Check leap year

~~Check leap year~~

```
    else
        printf("%d is not a leap year", year);
    }
else
    printf("%d is a leap year", year);
}
else
    printf("%d is not a leap year", year);

return 0;
}
```

```
Enter year: 6578
6578 is not a leap year
```

15.
```
#include <stdio.h>
int main()
{
 double num;
 printf("Enter a number: ");
 scanf("%lf", &num);
 if (num < 0.0)
    printf("You entered a negative number.");
 else if (num > 0.0)
    printf("You entered a positive number.");
 else
    printf("You entered 0.");
 return 0;
}
```

Check whether a number is positive or negative.

Check whether a number is positive or negative.

Enter a number : 6

You entered a positive number.

16.
```
#include<stdio.h>
int main(){
    char c;
    printf("Enter a character :");
    scanf("%c",&c);
    if((c>='a' && c<='z') || (c>='A' && c<='Z'))
        printf("%c is an alphabet.",c);
    else
        printf("%c is not an alphabet.",c);
    return 0;
}
```

Check whether a character is an alphabet

Enter a character : u

u is an alphabet.

17.
```
#include<stdio.h>
int main()
{
    int n,i,sum=0;
    printf("Enter a positive integer :");
    scanf("%",&n);
    for(i=1;i<=n;++i){
        sum+=i;
    }
```

Calculate the sum of natural numbers.

```
    printf("Sum = %d", sum);
    return 0;
}
```

☆ Calculate the sum of natural numbers

```
Enter a positive integer: 7
Sum = 28
```

18.
```
#include<stdio.h>
int main()
{
    int n, i;
    unsigned long long fact = 1;
    printf("Enter an integer : ");
    scanf("%d", &n);
    // shows error if the user enters a negative
    integer
    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;
}
```

Find factorial of a number

☆ Find factorial of a number

Enter an integer: 6
Factorial of 6 = 720

19.
```
#include <stdio.h>
int main()
{
  int n, i;
  printf("Enter an integer:");
  scanf("%d", &n);
 for (i=1; i<=10; ++i)
  {
    printf("%d *%d = %d \n", n, i, n*i);
  }
  return 0;
  }
```

Generate multiplication table

Enter an integer: 8
8 *1 =8
8 *2 = 16
8 *3 = 24
8 *4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
8 * 9 = 72
8 * 10 = 80

20.
```
#include<stdio.h>
int main ()
{
int i, n, t1=0, t2=1, next_term;
printf("Enter the number of terms: ");
scanf ("%d", &n);
printf ("Fibonacci series:");
for (i=1; i<=n; ++i)
{
printf ("%d, ", t1);
next_term = t1+t2;
t1 = t2;
t2 = next_term;
}
return 0;
}
```

Display fibonacci series

```
Enter the number of terms: 2
Fibonacci series : 0, 1.
```

21.
```
#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)
{
// check if i is a factor of n1 and n2.
```

Find GCD of two numbers

```
    if (n1 %i==0 & n2%i ==0)
        gcd =i;
}
printf("G.C.D. of %d and %d is %d", n1, n2, gcd);
return 0;
}
```

Enter two integers: 53 45
G.C.D. of 53 and 45 is 1.

22. Find LCM of two numbers

```
#include<stdio.h>
int main()
{
  int n1, n2, max;
  printf("Enter two positive integers: ");
  scanf("%d %d", &n1, &n2);
 //maximum number between n1 and n2 is
  stored in max
  max = (n1>n2) ? n1 : n2;
  while (1)
  {
    if (max %n1 ==0 && max %n2 == 0)
    {
      printf("The LCM of %d and %d is %d",
        n1, n2, max);
        break;
    }
    ++max;
  }
  return 0;
```

* Ternary → Condition ? do something if True : do something if False ;

Enter two positive integers : 5 16
The LCM of 5 and 16 is 80.

23. * Display characters from A to Z using loop

```
#include <stdio.h>
int main()
{
  char c;
  for (c = 'A' ; c <= 'Z' ; ++c)
     printf("%c", c);
  return 0;
}
```

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z.

24. * Count no. of digits as an integer

```
#include <stdio.h>
int main()
{
 long long n;
 int count = 0;
 printf("Enter an integer: ");
 scanf("%lld", &n);
 // iterate until n becomes 0
// remove last digit from n in each iteration.
// increase count by 1 in each iteration.
 while (n != 0)
 {
   n /= 10;
   ++count;
 }
```

```
printf("Number of digits : %d", count);
}
```

Enter an integer : 688
Number of digits : 3.

25.
```
#include <stdio.h>
int main()
{
    int n, rev = 0, remainder;
    printf("Enter an integer:");
    scanf("%d", &n);
    while (n != 0)
    {
      remainder = n % 10
      rev = rev * 10 + remainder;
      n /= 10;
    }
    printf("Reversed number = %d", rev);
    return 0;
}
```

* Reverse a number

Enter an integer : 678
Reversed number : 876

26.
```
#include <stdio.h>
int main()
{
    int base, exp;
    long long result = 1;
    printf("Enter a base number:");
    scanf("%d", &base);
    printf("Enter an exponent:");
    scanf("%d", &exp);
```

* Calculate the power of a number

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

Enter a base number : 7
Enter an exponent : 3

Answer = 343

27.
```
#include <stdio.h>
int main()
{
    int n, reversedN = 0, remainder, originalN;
    printf("Enter an integer:");
    scanf("%d", &n);
    originalN = n;

    // reversed integer is stored in reversedN.
    while (n != 0)
    {
        remainder = n % 10;
        reversedN = reversedN * 10 + remainder;
        n /= 10;

    // palindrome if originalN and reversedN are
        equal
    if (originalN == reversedN)
        printf("%d is a palindrome.", originalN);
```

* Check whether a number is palindrome or not.

```
    else
        printf("%d is not a palindrome.", originalN);
    return 0;
}
```

```
Enter an integer : 23432
23432 is a palindrome
```

☆

28.
```
#include<stdio.h>
int main()
{
  int n, i, flag=0;
  printf("Enter a positive integer:");
  scanf("%d", &n);
  for (i=2; i<=n/2; ++i){
    // condition for non-prime
      if (n%i==0){
        flag=1;
        break;
      }
  }
  if (n==1){
   printf("1 is neither prime nor composite.");
  }
  else{
     if (flag==0)
        printf("%d is a prime number.", n);
     else
        printf("%d is not a prime number.", n);
  }
  return 0;
```

☆ Check whether a no. is prime or not

Enter a positive integer : 6
6 is not a prime number.

29.
```
#include<stdio.h>
int main()
{
    int low, high, i, flag;
    printf("Enter two numbers (intervals) :");
    scanf("%d %d", &low, &high);
    printf("Prime numbers between %d and %d are:",
        low, high);

// iteration until low is not equal to high.
    while (low<high) {
        flag = 0;

        // ignore number less than 2
        if (low <= 1) {
            ++low;
            continue;
        }

        // if low is a non-prime number, flag will be 1.
        for (i=2; i <= low/2 ; ++i) {
            if (low % i ==0) {
                flag = 1;
                break;
            }
        }
```

Display prime number between two intervals

```
    if (flag == 0)
      printf ("%d", low);
      // to check prime for the next number
      // increase low by 1.
    ++low;
  }
  return 0;
}
```

Enter two numbers (intervals): 5 17
Prime numbers between 5 and 17 are: 5 7 11 13

30. ~~Check~~ Armstrong Number

```
#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 contains the last digit.
    remainder = originalNum % 10;
    result += remainder * remainder * remainder;
    // removing last digit from the original number
    originalNum /= 10;
  }
  if (result == num)
    printf ("%d is an Armstrong number.", num);
```

```
    else
        printf("%d is not an Armstrong number.", num);
    return 0;
}
```

Enter a three-digit ~~number~~ integer: 767
767 is not an Armstrong number.

31.
```
#include<math.h>
#include<stdio.h>
int main()
{
    int low, high, number, originalNumber, rem,
    count = 0;
    double result = 0.0;
    printf("Enter two numbers (intervals): ");
    scanf("%d %d", &low, &high);
    printf("Armstrong numbers between %d and
    %d are:", low, high);
// iterate number from (low+1) to (high-1)
// In each iteration, check if number is
                Armstrong.
    for (number = low+1; number < high; ++number)
    {
        originalNumber = number;
        // number of digits calculation
        while (originalNumber != 0) {
            originalNumber /= 10;
        } ++count;
```

★ Display Armstrong no. between two intervals

```
//result contains sum of nth power of
    individual digits.
while (originalNumber!=0){
    rem = originalNumber %10;
    result += pow(rem, count);
    originalNumber /=10;
}
/*check if number is equal to the sum of
nth power of individual digits */
    while(
if (int) result == number){
    printf ("%d", number);
}
//resetting the values
    count=0;
    result = 0;
}
return 0;
}
```

Enter two numbers (intervals): 343 444
Armstrong numbers between 343 and 444 are:
370 371 407

32). #include<stdio.h>

Display factors of a number

```
int main(){
    int num, i;
    printf("Enter a positive integer:");
    scanf("%d",&num);
    printf("Factors of %d are:", num);
    for(i=1; i<=num; ++i){
        if(num%i==0){
            printf("%d", i);
        }
    }

return 0;
}
```

Enter a positive number: 6
Factors of 6 are: 1 2 3 6

33). #include<stdio.h>

Print pyramids & patterns

```
int main(){
    int i, j, rows;
    printf("Enter the number of rows:");
    scanf("%d", &rows);
```

```
    for (i=1; i<=rows; ++i){
        for (j=1; j<=i; ++j){
            printf("* ");
        }
        printf("\n");
    }
    return 0;
}
```

```
Enter the number of rows: 3
*
* *
* * *
```

34. Print pyramids 4 Patterns

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

```
Enter the number of rows: 2
* *
*
```

```
#include<stdio.h>
int main(){
    int i, j;
    for(i=1; i<=5; ++i){
        for(j=i; j>=1; --j){
            printf("%d", j);
        }
        printf("\n");
    }
    return 0;
}
```

output:-

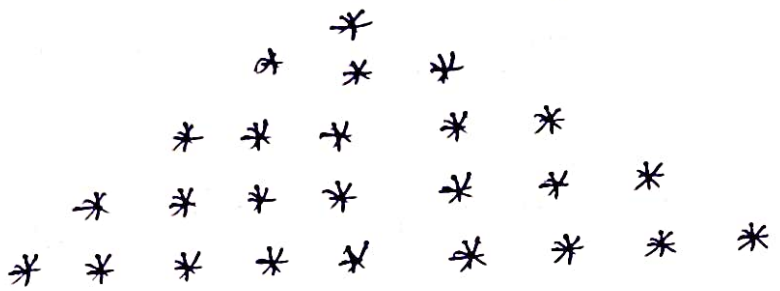
```
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
```

35) → Pyramids

```
#include<stdio.h>
int main(){
    int i, gap, rows, k=0;
    printf("Enter the number of rows:");
    scanf("%d", &rows);
    for(i=1; i<=rows; ++i, k=0){
        for(gap=1; gap<=rows-i; ++gap){
```

```
        printf("  ");
    }
    while (k!=2*i-1){
        printf("* ");
        ++k;
    }
    printf("\n");
}
return 0;
}
```

Enter the number of rows: 5



36. Pyramids

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

```
#include<stdio.h>
int main (){
    int i, j, k;
    for (i=1 ; i<=5 ; ++i){
        for (j=(5-i) ; j>=1; --j){
            printf ("  ");
            for (k=i ; k<=i ; ++k){
                printf ("* ");
            }
        }
        printf ("\n");
    }
    return 0;
}
```

(36) Output :-

```
Enter the number of rows! 5
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

37. * Swap two number without using 3rd variable

```
#include<stdio.h>
int main()
{
  int a,b;
  printf("Please enter any number:");
  scanf("%d", &a);
  printf("Please enter any number:");
  scanf("%d", &b);
  a=a+b;
  a=a-b;
  a=a-b;
  printf("After swapping, the numbers are %d %d", a,b);
  return 0;
}
```

- Nested conditional operator:-

→ Syntax:-

Condition ? [cond 2 ? _ : _ :] (TRUE) [cond 3 ? _ : _;] (FALSE)

38. * Find greatest among three no.

```
#include <stdio.h>
int main ()
{
  int a,b,c,d;
  printf("Enter three no.:");
  scanf("%d %d %d", &a, &b, &c);
  d = (a>b) ? a>c ? a : c : b>c ? b : c;
  printf("max is %d", d);
  return 0;
}
```

39. * Find greatest number among 6 numbers.

```
#include<stdio.h>
int main()
{
  int a,b,c,d,e,f, m1,m2,max;
  printf("Enter 6 numbers:");
  scanf("%d %d %d %d %d %d", &a,&b,&c,&d,&e,&f);
  m1 = (a>b)? (a>c)?a:c : (b>c)? b:c;
  m2 = (d>e)? (d>f)?d:f : (e>f)?e:f;
  max = (m1>m2)? m1:m2;
  printf("Max is %d", max);
  return 0;
}
```

40. * Display prime numbers between intervals using function.

```
#include<stdio.h>
int checkPrimeNumber(int,n);
int main()
{
  int n1,n2,i,flag;
  printf("Enter two positive integers");
  scanf("%d %d", &n1,&n2);
  printf("Prime numbers between %d and %d are:",
        n1,n2);
  for (i=n1+1; i<n2; ++i){
    // flag will be equal to 1 if i is prime
    flag = checkPrimeNumber(i);
    if (flag==1)
      printf("%d", i);
  }
```

```
    return 0;
}

// user-defined function to check prime number.
int checkPrimeNumber(int n){
    int j, flag=1;
    for(j=2; j<=n/2; ++j){
        if(n%j==0){
            flag=0;
            break;
        }
    }
    return 0;
}
```

```
Enter two positive integers: 67  98
Prime numbers between 67 and 98 are: 71  73
 79  83  89  97
```

41. Check Prime or Armstrong Number using User-defined function.

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

int checkPrimeNumber (int n);
int checkArmstrongNumber (int n);

int main () {
    int n, flag;
    printf (" Enter a positive integer: ");
    scanf ("%d", &n);

    // check prime number
    flag = checkPrimeNumber (n);
    if (flag == 1)
        printf ("%d is a prime number. \n", n);
    else
        printf ("%d is not a prime number \n", n);

    // Check Armstrong number
    flag = checkArmstrongNumber (n);
    if (flag == 1)
        printf ("%d is an Armstrong number.", n);
    else
        printf ("%d is not an Armstrong number.", n);
```

```
    return 0;
}

//function to check prime number
int checkPrimeNumber (int n) {
    int i, flag = 1, squareRoot;
    //computing the square root
    squareRoot = sqrt(n);
    for (i=2; i <= squareRoot; ++i) {
        //condition for non-prime number
        if (n % i == 0) {
            flag = 0;
            break;
        }
    }
    return flag;
}
//function to check Armstrong number
int checkArmstrongNumber (int num) {
    int originalNum, remainder, n=0, flag;
    double result = 0.0;

//store the number of digits of num in n.
    for (originalNum = num; originalNum != 0; ++n){
        originalNum /= 10;
    }
for (originalNum = num; originalNum != 0; originalNum /= 10){
        remainder = originalNum % 10;
```

```
/* store the sum of the power of individual
   digits in result */
    result += pow(remainder, n);
    // condition for Armstrong Number
    if (round(result) == num)
        flag = 1;
    else
        flag = 0;
    return flag;
}
```

Enter a positive integer: 56
56 is not a prime number.
56 is not an Armstrong number.

42) Check whether a no. can be expressed as sum of two prime no.

```
#include <stdio.h>
int checkPrime(int n);
int main() {
    int n, i, flag = 0;
    printf("Enter a positive integer: ");
    scanf("%d", &n);
    for (i = 2; i <= n/2; ++i) {
    // condition for i to be a prime number
    if (checkPrime(i) == 1) {
    // condition for n-i to be a prime number
        if (checkPrime(n-i) == 1) {
```

```
            printf("%d = %d + %d \n", n, i, n-i);
            flag = 1;
        }
    }
    if (flag == 0)
        printf("%d cannot be expressed as the sum
    of two prime numbers.", n);
    return 0;
}

//function to check prime number
int checkPrime (int n){
    int i, isPrime = 1;
    for (i = 2; i <= n/2; ++i){
        if (n % i == 0){
            isPrime = 0;
            break;
        }
    }
    return isPrime;
}
```

Enter a positive integer : 67

67 cannot be expressed as the sum of two prime numbers.

43). # include <stdio.h>
int addNumbers (int n);

```
int main() {
    int num;
    printf("Enter a positive integer: ");
    scanf("%d", &num);
    printf("sum = %d", addNumbers(num));
    return 0;
}

int addNumbers(int n) {
  if (n!=0)
    return n + addNumbers(n-1);
  else
    return n;
}
```

* Find the sum of natural no. using Recursion

Enter a positive number/integer: 76

Sum = 2926.

44)

```
#include<stdio.h>
long int multiplyNumbers(int n);
int main() {
    int n;
    printf("Enter a positive integer:");
    scanf("%d", &n);
    printf("Factorial of %d = %ld", n,
           multiplyNumbers(n));
    return 0;
}
```

* Find factorial of a number using recursion

```
long int multiplyNumbers (int n){
  if (n>=1)
    return n* multiplyNumbers (n-1);
  else
    return 1;
}
```

Enter a positive integer : 5

Factorial of 5 = 120.

45) Find G.C.D. using Recursion

```
#include <stdio.h>
int hcf(int n1, int n2);
int main (){
    int n1, n2;
    printf ("Enter two positive integers : ");
    scanf ("%d %d", &n1, &n2);
    printf ("G.C.D. of %d and %d is %d.",
        n1, n2, hcf(n1,n2));
    return 0;
}
int hcf (int n1, int n2){
  if (n2 != 0)
    return hcf (n2, n1%n2);
  else
    return n1;
}
```

Enter two positive integers : 15 45

G.C.D. of 15 and 45 is 15.

46) #include <math.h>
#include <stdio.h>
int convert(long long n);
int main() {
long long n;
printf("Enter a binary number: ");
scanf("%lld", &n);
printf("%lld in binary = %d in decimal",
n, convert(n));
return 0;
}
int convert(long long n) {
int dec = 0, i = 0, rem;
while (n!=0) {
rem = n % 10;
n /= 10;
dec += rem * pow(2, i);
++i;
}
return dec;
}

* convert Binary to decimal

Enter a binary number: 1100
1100 in binary = 12 in decimal

47) #include <math.h>
#include <stdio.h>
long long convert(int n);

Convert Decimal into Binary

```
int main () {
    int n;
    printf("Enter a decimal number :");
    scanf ("%d", &n);
    printf (" %d in decimal = %lld in binary", n,
            convert (n));
    return 0;
}
long long convert (int n) {
    long long bin =0;
    int rem, i =1, step =1;
    while (n!=0) {
        rem =n%2;
        printf ("step %d : %d/2 , Remainder = %d,
            Quotient = %d \n" , step ++, n, rem, n/2);
        n/=2;
        bin +=rem * i;
        i *=10;
    }
    return bin;
}
```

```
Enter a decimal number :12
step 1: 12/2 , Remainder =0, Quotient = 6
step 2: 6/2 , Remainder = 0, Quotient = 3
step 3: 3/2 , Remainder =1, Quotient =1
step 4: 1/2 , Remainder = 1, Quotient =0
12 in decimal = 1100 in binary
```

48) convert decimal to octal

```
#include <math.h>
#include <stdio.h>
int convertDecimalToOctal(int decimalNumber);
int main()
{
  int decimalNumber;
  printf("Enter a decimal number : ");
  scanf("%d", &decimalNumber);
  printf(" %d in decimal = %d in octal ",
        decimalNumber,
  convertDecimalToOctal(decimalNumber));
  return 0;
}
int convertDecimalToOctal(int decimalNumber){
  int octalNumber = 0, i = 1;
  while (decimalNumber != 0){
    octalNumber += (decimalNumber % 8) * i;
    decimalNumber /= 8;
    i *= 10;
  }
  return octalNumber;
}
```

Output :-

Enter a decimal number : 45

45 in decimal = 55 in octal

49). #include <stdio.h> ✱ Calculate Average using Arrays.

```
int main(){
    int n, i;
    double num[100], sum=0.0, avg;

    printf("Enter the number of elements:");
    scanf("%d", &n);

    while (n>100 || n<1){
        printf("Error! number should in range of
          (1 to 100) .\n");
        printf("Enter the number again:");
        scanf(" %d", &n);
    }
    for(i=0; i<n; ++i){
```

```
    printf("%d. Enter number: ", i+1);
    scanf("%lf", &num[i]);
    sum += num[i];
  }
  avg = sum/n;
  printf("Average = %.2lf", avg);
  return 0;
}
```

Output:-

```
Enter the number of elements: 5
1. Enter number: 4
2. Enter number: 6
3. Enter number: 7
4. Enter number: 80
5. Enter number: 67
Average = 32.80
```

* Find largest elements in an Array:-

```
#include <stdio.h>
int main(){
    int i, n;
    double arr[100];
    printf("Enter the number of element (1 to 100): ");
    scanf("%lf", &arr[i]);

    for (i=0; i<n; ++i){
        printf("Enter number %d: ", i+1);
        scanf("%lf", &arr[i]);
    }
```

```
//storing the largest number to arr[0]
for (i=1; i<n ; ++i) {
    if (arr[0] < arr[i])
        arr[0] = arr[i]
}

printf(" Largest element = %.2lf ", arr[0]);

return 0;
}
```

Output :-

```
Enter the number of elements (1 to 100): 6
Enter number 1 : 6
Enter number 2 : 9
Enter number 3 : 7
Enter number 4 : 67
Enter number 5 : 89
Enter number 6 : 79

Largest element = 89.00
```

* Calculate standard deviation

```
#include<stdio.h>
#include<math.h>
//function to compute standard deviation.
double calculateSD (double data[]) {
    double sum=0.0, mean, SD=0.0;
    int i;
    for (i=0; i<10; ++i) {
        sum += data[i];
    }
```

```
    mean = sum/10;
    for (i=0; i<10; ++i) {
        sd += pow(data[i] - mean, 2);
    }
    return sqrt(sd/10);
}

int main() {
    int i;
    double data[10];
    // take inputs from user
    printf("Enter 10 elements :");
    for (i=0; i<10; ++i){
        scanf("%lf", &data[i]);
    }
    printf("\n Standard Deviation = %.6lf",
        calculate sd(data));
    return 0;
}
```

Output:-

Enter 10 Elements : 4 5 6 5 67 87 98 4 3 6

Standard Deviation = 37.016888

* Add two matrices using multi-dimensional Array:-

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

```
int r,c, a[100][100], b[100][100], sum[100][100], i, j;
printf("Enter the number of rows (between 1 and 100):");
scanf("%d", &r);
printf("Enter the number of columns
  (between 1 and 100):");
scanf("%d", &c);
printf("\nEnter elements of 1st elements
   matrix:\n");
for (i=0; i<r; ++i){
    for (j=0; j<c; ++j){
      printf("Enter elements a%d%d:",
          i+1, j+1);
      scanf("%d", &a[i][j]);
    }
}
printf("Enter elements of 2nd matrix:\n");
for(i=0; i<r; ++i){
    for(j=0; j<c; ++j){
        printf("Enter element a%d%d:",
            i+1, j+1);
        scanf("%d", &b[i][j]);
    }
}
// adding two matrices
for (i=0; i<r; ++i)
    for (j=0; j<c; ++j){
```

```
            sum[i][j] = a[i][j] + b[i][j];
        }

//printing("\n Sum of two matrices: \n");
for

//printing the result
printf("\n Sum of two matrices: \n");
for (i=0; i<r; ++i){
    for (j=0; j<c; ++j){
        printf("%d  ", sum[i][j]);
        if(j==c-1){
            printf("\n\n");
        }
    }
}

return 0;
}
```

Output:-

```
Enter the number of rows (between 1 and 100): 3
Enter the number of columns (between 1 & 100): 3

Enter elements of 1st matrix:
Enter element a11: 2
Enter element a12: 3
Enter element a13: 6
Enter element a21: 2
Enter element a22: 2
Enter element a23: 2
Enter element a31: 2
Enter element a32: 2
Enter element a33: 2
```

```
Enter elements of 2nd matrix:
Enter element a11 : 3
Enter element a12 : 3
Enter element a13 : 3
Enter element a21 : 3
Enter element a22 : 3
Enter element a23 : 3
Enter element a31 : 3
Enter element a32 : 3
Enter element a33 : 3

Sum of two matrices:
5    6    9
5    5    5
5    5    5
```

53. Multiply two matrices using multi-dimensional Arrays.

```
#include<stdio.h>
//function to get matrix elements entered by the user
void getMatrixElements(int matrix[][10], int row, int column){
    printf("\nEntered elements\n");
    for(int i=0; i<row; ++i){
        for(int j=0; j<column; ++j){
            printf("Enter a%d%d:", i+1, j+1);
            scanf("%d", &matrix[i][j]);
        }
    }
}
```

```
//function to multiply two matrices
void multiply_matrices (int first[][10], int second[][10],
        int result[][10], int r1, int c1, int r2, int c2){

  // initialising elements of matrix mult to 0.
  for (int i=0; i<r1; ++i){
     for (int j=0; j<c2; ++j){
        result[i][j]=0;
     }
  }

  /*multiplying first and second matrices and
    storing it in result */
  for (int i=0; i<r1; ++i){
     for(int j=0; j<c2; ++j){
        for(int k=0; k<c1; ++k){
           result[i][j] += first[i][k] * second[k][j];
        }
     }
  }
}

//function to display the matrix
void display (int result[][10], int row, int column){
     printf("\nOutput Matrix:\n");
     for (int i=0; i<row; ++i){
        for (int j=0; j<column; ++j){
           printf("%d", result[i][j]);
           if (j==column-1)
              printf("\n");
```

```
    }
  }
}

int main() {
  int first[10][10], second[10][10], result[10][10],
      r1, c1, r2, c2;
  printf("Enter rows and columns for the first matrix: ");
  scanf("%d %d", &r1, &c1);
  printf("Enter rows and columns for the second matrix: ");
  scanf("%d %d", &r2, &c2);

// Taking input until
// 1st matrix column is not equal to 2nd matrix row
while (c1 != r2) {
    printf("Error! Enter rows and columns again.\n");
    printf("Enter rows and columns for the first
        matrix: ");
    scanf("%d %d", &r1, &c1);
    printf("Enter rows and columns for the second
        matrix: ");
    scanf("%d %d", &r2, &c2);
  }

// get elements of a first matrix
getMatrixElements(first, r1, c1);

// get elements of a second matrix
```

```
getmatrixElements(second, r2, c2);

//multiply two matrices
multiply_matrices (first, second, result, r1, c1, r2, c2);

//display (result, r1, r2);

return 0;
}
```

Output :-

```
Enter rows and columns of the first matrix: 2 2
Enter rows and colums of the second matrix: 2 2

Enter Elements:
Enter a11: 2
Enter a12: 4
Enter a21: 3
Enter a22: 5

Enter elements:
Enter a11: 3
Enter a12: 1
Enter a21: 2
Enter a22: 5

Output matrix:
12    17
22    29
```

54% ✱ Find a transpose of matrix.

```
#include<stdio.h>
int main(){
    int a[10][10], transpose[10][10], r, c, i, j;
    printf("Enter rows and columns:");
    scanf ("%d %d", &r, &c);

  // Assigning elements to the matrix.
    printf("\n Elem- Enter matrix elements:\n");

    for (i=0; i<r ; ++i)
      for(j=0; j<c ; ++j){
        printf("Enter element a %d%d: ", i+1, j+1);
        scanf ("%d", &a[i][j]);
      }

// Displaying the matrix a[][]
printf("\nEntered matrix :\n");
for (i=0; i<r ; ++i)
    for(j=0; j<c; ++j){
        printf("%d ", a[i][j]);
        if (j==c-1)
          printf("\n");
    }
// Find the transpose of matrix a
 for(i=0 ; i<r; ++i)
    for (j=0 ; j<c ; ++j){
        transpose[j][i] = a[i][j];
    }
```

```
//Displaying the the transpose of matrix a
printf("\n Transpose of the matrix:\n");
for (i=0; i<c ; ++i)
    for (j=0; j<r ; ++j){
        printf("%d  ", transpose[i][j]);
        if (j==r-1)
            printf("\n");
    }
  return 0;
}
```

Output:-

```
Enter rows and columns: 3 4
Enter matrix elements:
Enter element a11: 2
Enter element a12: 3
Enter element a13: 2
Enter element a14: 5
Enter element a21: 5
Enter element a22: 6
Enter element a23: 4
Enter element a24: 1
Enter element a31: 2
Enter element a32: 5
Enter element a33: 4
Enter element a34: 6
```

Entered matrix:

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

Transpose of the matrix:

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

55) Multiply two matrices by passing matrix to a function.

```
#include<stdio.h>
void enterData(int firstMatrix[][10], int secondMatrix[][10
 int rowfirst, int columnfirst, int rowSecond,
 int columnSecond);

void multiplyMatrices(int firstMatrix[][10],
      int secondMatrix[][10], int multResult[][10],
      int rowfirst, int columnfirst, int rowsecond,
      int columnsecond);

void display(int mult[][10], int rowfirst,
             int columnSecond);

int main(){
  int firstMatrix[10][10], secondMatrix[10][10],
      mult[10][10], rowfirst, columnfirst, rowSecond,
      columnSecond, i, j, k;
```

```
printf("Enter rows and colums for first matrix:");
scanf("%d %d", &rowfirst, &columnfirst);
printf("Enter rows and columns for second matrix:");
scanf("%d %d", &rowsecond, &columnSecond);

/* if column of first matrix is not equal to row
of second matrix, asking user to enter the
size of matrix again */
while (columnfirst != rowsecond) {
    printf("Error! column of first matrix not equal
    to row of second. \n");
    printf("Enter rows and column for first
        matrix:");
    scanf("%d %d", &rowfirst, &columfirst);
    printf("Enter rows and columns for second
        matrix:");
    scanf("%d %d", &rowsecond, &columnsecond);
}

// Function to take matrices data.
enterData(firstMatrix, secondMatrix, rowfirst,
        columnfirst, rowsecond, Columnsecond);

// Function to multiply two matrices.
multiplyMatrices(firstmatrix, secondMatrix, mult,
        rowfirst, columnfirst, rowsecond, columnsecond);
/*Function to display resultant matrix after
multiplication*/
```

```
    display (mult , rowfirst, columnSecond);
    return 0;
}
void enterData(int firstMatrix [ ][10], int secondMatrix[ ][1
    int rowfirst, int columnfirst, int rowSecond,
    int columnsecond) {

    int i, j;
    printf("\n Enter elements of matrix 1: \n");
    for(i=0; i < rowfirst ; ++i) {
        for (j=0; j < columnfirst; ++j) {
            printf("Enter element a %d %d: ", i+1, j+1);
            scanf("%d", &firstMatrix[i][j] );
        }
    }
    printf("\n Enter elements of matrix 2: \n");
    for (i=0; i < rowsecond ; ++i) {
        for (j=0; j <columnsecond ; ++j) {
            printf("Enter element b %d %d: ", i+1,
                j+1);
            scanf("%d", &secondMatrix[i][j]);
        }
    }
}
void multiplyMatrices (int firstMatrix[ ][10], int secondmatrix[ ]
[10], int mult[ ][10], int rowfirst, int columnfirst,
    int rowSecond, int columnSecond) {
```

```
    int i, j, k;
// initializing elements of matrix mult to 0.
    for (i=0; i< rowfirst ; ++i){
        for (j=0; j< columnsecond ; ++j){
            mult[i][j]=0;
        }
    }
/* multiplying matrix firstmatrix and secondMatrix
   and storing in array mult */
    for (i=0; i< rowfirst; ++i){
        for (j=0 ; j < columnsecond ; ++j){
            for (k=0; k< columnfirst;  ++k){
                mult [i][j] += firstMatrix[i][k] *
                               secondMatrix[k][j];
            }
        }
    }
}
void display(int mult[][10], int rowfirst, int
             columnSecond){
    int i,j;
    printf("\nOutput Matrix:\n");
    for (i=0 ; i < rowfirst ; ++i){
        for(j=0 ; j< columnSecond ; ++j){
            printf("%d", mult[i][j]);
            if (j== columnsecond-1)
                printf("\n\n");
        }
```

55. ✱ Convert Octal to Decimal

```
#include<stdio.h>
#include<math.h>
long long convertOctalToDecimal(int octalNumber);
int main(){
    int octalNumber;
    printf("Enter an octal number:");
    scanf("%d",&octalNumber);
    printf("%d in octal = %lld in decimal",
            octalNumber,
    convertOctalToDecimal(octalNumber));

    return 0;
}
long long convertOctalToDecimal(int octalNumber){
    int decimalNumber = 0, i=0;
    while(octalNumber!=0){
        decimalNumber += (octalNumber%10)*pow(8,i);
        ++i;
        octalNumber/=10;
    }
    i=1;
    return decimalNumber;
}
```

Output:-

Enter an octal number: 65
65 in octal = 53 in decimal

57) Convert binary to octal.

```
#include <math.h>
#include <stdio.h>
int convert(long long bin);
int main(){
    long long bin;
    printf("Enter a binary number:");
    scanf("%lld", &bin);
    printf("%lld in binary = %d in octal", bin,
           convert(bin));
    return 0;
}
int convert(long long bin){
    int oct=0, dec=0, i=0;
//converting binary to decimal
while (bin!=0){
    dec += (bin%10) * pow(2,i);
    ++i;
    bin/=10;
}
i=1;
//converting to decimal to octal
while (dec!=0){
    oct += (dec%8) * i;
    dec /=8;
    i *=10;
}
return oct;
}
```

Output:-

```
Enter a binary number: 11
11 in binary = 3 in octal.
```

58). Convert octal to binary

```
#include<math.h>
#include <stdio.h>
long long convert(int oct);
int oct;
printf("Enter an octal number:");
scanf("%d", &oct);
printf("%d in octal =%lld in binary", oct,
        convert(oct));
return 0;
}

long long convert(int oct){
    int dec=0, i=0;
    long long bin=0;
    //converting octal to decimal
    while (oct!=0){
        dec += (oct%10) * pow(8,i);
        ++i;
        oct/=10;
    }
    i=1;
    //converting decimal to binary
    while (dec!=0){
        bin += (dec%2)*i;
```

```
        dec /=2;
        i *=10;
    }
    return bin;
}
```

Output:-

Enter an octal number:

65 in octal = 110101 in binary.

59. Reverse a sentence using Recursion.

```
#include<stdio.h>
void reverseSentence();
int main(){
    printf("Enter a sentence: ");
    reverseSentence();
    return 0;
}

void reverseSentence(){
    char c;
    scanf("%c", &c);
    if(c!='\n'){
        reverseSentence();
        printf("%c",c);
    }
}
```

Output:-

Enter a sentence: gharnijana

anajinrahg

60). Calculate the power using recursion

```
#include<stdio.h>
int power (int n1, int n2);
int main (){
    int base, a, result;
    printf ("Enter base number : ");
    scanf ("%d", &base);
    printf ("Enter power number : (positive
            integer) ");
    scanf ("%d", &a);
    result = power (base, a);
    printf ("%d ^ %d = %d", base, a, result);
    return 0;
}
int power (int base, int a){
    if (a!=0)
        return (base * power (base, a-1));
    else
        return 1;
}
```

Output!-

Enter base number : 2

Enter power number (positive integer) : 5

2 ^ 5 = 32

61) ☆ Access Array Elements using pointer.

```
# include <stdio.h>
int main() {
   int data[5];
   printf ("Enter element :");
   for (int i = 0; i<5 ; ++i){
      scanf("%d", data+i);
   }
   printf (" You entered : \n");
   for (int i=0 ; i<5 ; ++i) {
      printf (" %d \n", *(data +i));
   }
   return 0;
}
```

Output!-

Enter elements : 2 3 4 5 6
You entered:
2
3
4
5
6

62) ☆ Swap numbers in Cyclic Order using call by Reference.

```
#include <stdio.h>
void cyclicSwap (int *a, int *b, int *c);
int main () {
    int a, b, c ;
    printf ("Enter a, b and c respectively : ");
```

```
    scanf("%d %d %d", &a, &b, &c);
    printf("value before swapping :\n");
    printf("a= %d, \n b= %d \n c= %d\n", a, b, c);
    cyclicswap (&a, &b, &c);
    printf("Value after swapping :\n");
    printf("a = %d \n b = %d \n c = %d", a, b, c);
    return 0;
}

void cyclicSwap (int *n1, int *n2, int *n3){
    int temp;
// swapping in cyclic order.
    temp = *n2;
    *n2 = *n1;
    *n1 = *n3;
    *n3 = temp;
}
```

Output:-

Enter a, b and c respectively : 4 6 8

Value Before swapping:

a = 4

b = 6

c = 8

Value After swapping!

a = 8

b = 4

c = 6

62). ✱ Find Largest number using Dynamic memory allocation

calloc , stdlib, are not in syllabus.

---

63). ✱ Store information of a student using structure

```
#include<stdio.h>
struct student {
    char name[50];
    int roll;
    double marks;
} s;
int main(){
    printf(" Enter information : \n");
    printf ("Enter name :");
    fgets ( s.name , sizeof (s.name ), stdin);
    printf (" Enter roll number : ");
    scanf ("%d", &s.roll );
    printf (" Enter marks :");
    scanf ("%lf", &s. marks);
    printf (" Displaying information :\n");
    printf (" Name :");
    printf ("%s ", s.name);
    printf ("Roll number : %d \n", s.roll);
    printf (" Marks : %.2lf \n", s. marks);
    return 0;
}
```

Output!-

```
Enter information!
Enter name: sj
Enter roll number: 45
Enter marks: 11

Displaying Information!
Name: Sj
Roll number: 45
Marks: 11.0.
```

644. Add Two Distance (in inch - feet system) using structure

```
#include<stdio.h>
struct Distance {
    int feet, inch;
} d1, d2, result;
int main(){
    printf("Enter 1st Distance \n");
    printf("Enter feet:");
    scanf("%d", &d1.feet);
    printf("Enter inch:");
    scanf("%d", &d1.inch);
    printf("Enter 2nd Distance \n");
    printf("Enter feet:");
    scanf("%d", &d2.feet);
    printf("Enter inch:");
```

```
    scanf("%d", &d2.inch);

    result.feet = d1.feet + d2.feet;
    result.inch = d1.inch + d2.inch;

    printf"
     result.feet += result.inch / 12;
     result.inch = result.inch % 12;
     printf("\n Sum of distances = %d \'- %d\" ",
      result.feet, result.inch);
      return 0;
    }
```

Output:-

```
Enter 1st Distance:
 Enter feet : 5
Enter inch : 6
 Enter 2nd Distance:
  Enter feet : 7
  Enter inch : 9

   Sum of distances = 13'-3"
```

65). ~~**~~ Add two complex numbers by passing structure to a function.

* we use the typedef to create an alias for a structure in order to simplify the declaration of structure variable.

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

```
typedef struct complex{
    double real;
    double imag;
    } complex;
complex add (complex n1, complex n2);
int main (){
    complex n1, n2, result;
    printf("For 1st complex number \n");
    printf ("Enter the real and imaginary parts:");
    scanf ("%lf %lf ", &n1.real, &n1.imag);
    printf ("\n for 2nd complex number \n");
    scanf
    printf ("Enter the real & imaginary parts:");
    scanf (" %lf %lf", &n2.real, &n2.img);
    result = add(n1, n2);
    printf ("Sum = %.1lf + %.1lfi", result.real,
        result.img);
    return 0;
}
complex add (complex n1, complex n2){
    complex temp;
    temp.real = n1.real + n2.real;
    temp.img = n1.imag + n2.img;
    return (temp);
}
```

Output!-

```
for 1st complex number
Enter the real and imaginary parts: 5 6
for 2nd complex number
Enter the real and imaginary parts: 7 9

Sum = 12.0+15.0i
```

66. Calculate difference between two time periods.

```
#include<stdio.h>
struct TIME{
    int seconds;
    int minutes;
    int hours;
};
void differencebetweentimeperiod(struct TIME t1,
     struct TIME t2, struct TIME * diff);

int main(){
    struct TIME startTime, stopTime, diff;
    printf("Enter the start time.\n");
    printf("Enter hours, minutes and seconds:");
    scanf("%d %d %d", &startTime.hours, &startTime.minutes,
          &startTime.seconds);
    printf("Enter the stop time:\n");
    printf("Enter hours, minutes and seconds:");
```

```
scanf("%d %d %d", &stopTime.hours, &stopTime.minutes,
        &stopTime.seconds);

// Difference between start & stop time
differenceBetweenTimePeriod(startTime, stopTime, &diff);
printf("\n Time Difference : %d : %d : %d - ",
    startTime.hours, startTime.minutes, startTime.seconds);
    printf("%d:%d:%d ", stopTime.hours, stopTime.minutes,
        stopTime.seconds);
    printf("= %d: %d: %d \n", diff.hours, diff.minutes,
        diff.seconds);

    return 0;
}

// Computes difference between time periods.
void differenceBetweenTimePeriod(struct TIME start,
    struct TIME stop, struct TIME *diff) {
    while (stop.seconds > start.seconds) {
        --start.minutes;
        start.seconds += 60;
    }
    diff->seconds = start.seconds - stop.seconds;
    while (stop.minutes > start.minutes) {
        --start.hours;
        start.minutes += 60;
    }
    diff->minutes = start.minutes - stop.minutes;
    diff->hours = start.hours - stop.hours;
}
```

Output :-

```
Enter the start time.
Enter hours, minutes, and seconds : 0 0 0
Enter the stop time.
Enter hours , minutes and seconds : 8 9 6

Time Difference : 0:0:0 - 8:9:6 = -9:50:54
```

67) Store information of student using structure.

```
#include<stdio.h>
struct student {
    char firstName[50];
    int roll;
    double marks;
} s[5];
int main(){
    int i;
    printf("Enter information of students:\n");
    // storing information.
    for(i=0; i<5; ++i){
        s[i].roll = i+1;
        printf("\nFor roll number %d,\n", s[i].roll);
        print("Enter first name:");
        scanf("%s", s[i].firstName);
        printf("Enter marks :");
        scanf("%f", &s[i].marks);
    }
```

```
printf("Displaying Information:\n\n");
// Displaying information
for (i=0; i<5 ; ++i)<
    printf("\nRoll number : %d\n", i+1);
    printf(" first name : ");
    puts(s[i].firstName );
    printf("marks : %.1f", s[i].marks);
    printf("\n");
}
return 0;
}
```