THIS BOOK CONSISTS OF AS MORE AS POSSIBLE C-PROGRAMS THAT ARE CONTAINING ALL BASICS AND THOSE ARE REQUIRED FOR INTERVIEW THIS MIGHT BE HELPFUL IN MANY WAYS TO CRACK CODING TEST OR TECHNICAL ROUND INTERVIEW . ~K V S NAIDU

THIS DOCUMENT CONTAINS AS FOLLOWS

- **Frequently asked c programs in interview**

C program with numbers

1. Write a c program to reverse any number.
2. Write a c program to find out sum of digit of given number.
3. Write a c program to find out power of number.
4. Write a c program to add two numbers without using addition operator.
5. Write a c program to subtract two numbers without using subtraction operator.
6. Write a c program to find largest among three numbers using binary minus operator.
7. Write a c program to find largest among three numbers using conditional operator
8. Write a c program to find out generic root of any number.
9. Write a c program to find out prime factor of given number.
10. Write a c program to find out NCR factor of given number.
11. How to convert string to int without using library functions in c
12. Program in c to print 1 to 100 without using loop
13. C program for swapping of two numbers
14. Program to find largest of n numbers in c
15. Split number into digits in c programming
16. C program to count number of digits in a number

Recursion

Example of recursion in c programming

L.C.M and H.C.F.

1. Write a c program to find out L.C.M. of two numbers.
2. Write a c program to find out H.C.F. of two numbers.

3. Write a c program to find out G.C.D. of two numbers.

Swapping

1. Write a c program to swap two numbers.
2. Write a c program to swap two numbers without using third variable.
3. Write a c program for swapping of two arrays.
4. Write a c program for swapping of two string.

Conversion (Number System)

1. Write a c program to convert decimal number to binary number.
2. Write a c program to convert decimal number to octal number.
3. Write a c program to convert decimal number to hexadecimal number.
4. Write a c program to convert octal number to binary number.
5. Write a c program to convert octal number to decimal number.
6. Write a c program to convert octal number to hexadecimal number.
7. Write a c program to convert hexadecimal number to binary number.
8. Write a c program to convert hexadecimal number to octal number.
9. Write a c program to convert hexadecimal number to decimal number.
10. Write a c program to convert binary number to octal number.
11. Write a c program to convert binary number to decimal number.

12. Write a c program to convert binary number to hexadecimal number.
13. C program for addition of binary numbers .
14. C program for multiplication of two binary numbers.
15. C program fractional binary conversion from decimal.
16. C program for fractional decimal to binary fraction conversion.
17. C program to convert decimal number to roman.
18. C program to convert roman number to decimal number.
19. C program to convert each digits of a number in words
20. C program to convert currency or number in word.

Conversion (Unit)

1. C program for unit conversion.

String

1. Write a c program to convert the string from upper case to lower case.
2. Write a c program to convert the string from lower case to upper case.
3. Write a c program to delete the all consonants from given string.
4. Write a c program to count the different types of characters in given string.
5. Write a c program to sort the characters of a string.
6. Write a c program for concatenation two strings without using string.h header file.
7. Write a c program to find the length of a string using pointer.
8. Write a c program which prints initial of any name.

9. Write a c program to print the string from given character.
10. Write a c program to reverse a string
11. Reverse a string using recursion in c
12. String concatenation in c without using strcat
13. How to compare two strings in c without using strcmp
14. String copy without using strcpy in c
15. Convert a string to ASCII in c

Matrix

1. Write a c program for addition of two matrices.
2. Write a c program for subtraction of two matrices
3. Write a c program for multiplication of two matrices.
4. Write a c program to find out sum of diagonal element of a matrix.
5. Write a c program to find out transport of a matrix.
6. Write a c program for scalar multiplication of matrix.
7. C program to find inverse of a matrix
8. Lower triangular matrix in c
9. Upper triangular matrix in c
10. Strassen's matrix multiplication program in c
11. C program to find determinant of a matrix

File

1. Write a c program to open a file and write some text and close its.
2. Write a c program to delete a file.
3. Write a c program to copy a file from one location to other location.
4. Write a c program to copy a data of file to other file.
5. Write a c program which display source code as a output.

6. Write a c program which writes string in the file.
7. Write a c program which reads string from file.
8. Write a c program which writes array in the file.
9. Write a c program which concatenate two file and write it third file.
10. Write a c program to find out size of any file.
11. Write a c program to know type of file.
12. Write a c program to know permission of any file.
13. Write a c program to know last date of modification of any file.
14. Write a c program to find size and drive of any file.

Complex number

1. Complex numbers program in c
2. Write a c program for addition and subtraction of two complex numbers.
3. Write a c program for multiplication of two complex numbers.
4. Write a c program for division two complex numbers.

Series

1. Write a c program to find out the sum of series 1 + 2 + …. + n.
2. Write a c program to find out the sum of series 1^2 + 2^2 + …. + n^2.
3. Write a c program to find out the sum of series 1^3 + 2^3 + …. + n^3.
4. Write a c program to find out the sum of given A.P.
5. Write a c program to find out the sum of given G.P.
6. Write a c program to find out the sum of given H.P.

7. Write a c program to find out the sum of series 1 + 2 + 4 + 8 … to infinity.

Array

1. Write a c program to find out largest element of an array.
2. Write a c program to find out second largest element of an unsorted array.
3. Write a c program to find out second smallest element of an unsorted array.
4. Write a c program which deletes the duplicate element of an array.
5. Write a c program for delete an element at desired position in an array.
6. Write a c program for insert an element at desired position in an array.
7. C program to find largest and smallest number in an array

Sorting

1. Write a c program for bubble sort.
2. Write a c program for insertion sort.
3. Write a c program for selection sort.
4. Write a c program for quick sort.
5. Write a c program for heap sort.
6. Write a c program for merge sort.
7. Write a c program for shell sort.

Recursion

1. Write a c program to find factorial of a number using recursion.

2. Write a c program to find GCD of a two numbers using recursion.
3. Write a c program to find out sum digits of a number using recursion.
4. Write a c program to find power of a number using function recursion.
5. Write a c program to reverse any number using recursion.

Size of data type

1. Write a c program to find the size of int without using sizeof operator.
2. Write a c program to find the size of double without using sizeof operator.
3. Write a c program to find the size of structure without using sizeof operator.
4. Write a c program to find the size of union without using sizeof operator.

Using pointer

1. Write a c program for concatenation two string using pointer.

Searching

1. Write a c program for linear search.
2. Write a c program for binary search.
3. Write a c program for binary search using recursion.

Area and volume

1. Write a c program to find the area of circle.

2. Write a c program to find the area of any triangle.
3. Write a c program to find the area of equilateral triangle.
4. Write a c program to find the area of right angled triangle.
5. Write a c program to find the area of rectangle.
6. Write a c program to find the area of trapezium.
7. Write a c program to find the area of rhombus.
8. Write a c program to find the area of parallelogram.
9. Write a c program to find the volume and surface area of cube.
10. Write a c program to find the volume and surface area of cuboids.
11. Write a c program to find the volume and surface area of cylinder.
12. Write a c program to find the surface area and volume of a cone.
13. Write a c program to find the volume and surface area of sphere.
14. Write a c program to find the perimeter of a circle, rectangle and triangle.

C program with very large numbers

1. Write a c program to find factorial of 100 or very large numbers
2. Write a c program to multiply the two very large number (larger the long int)
3. Write a c program for division of large number (larger than long int)
4. C code for modular division of large number.
5. C code for division of large number.
6. C code for power of large numbers.

Other

1. C program for ATM transaction.

2. Write a c program which passes one dimension array to function.
3. Write a c program which passes two dimension array to function.
4. Write a c program which takes password from user.
5. Write a scanf function in c which accept sentence from user.
6. Write a scanf function in c which accept paragraph from user.
7. Write a c program to print the all prime numbers between 1 to 300.
8. Write a c program which passes structure to function.
9. Palindrome in c without using string function
10. How to get the ASCII value of a character in c
11. C program to get last two digits of year
12. C program without main function

PERFECT NUMBER

Definition of perfect number or What is perfect number?

Perfect number is a positive number which sum of all positive divisors excluding that number is equal to that number. For example 6 is perfect number since divisor of 6 are 1, 2 and 3. Sum of its divisor is
1 + 2+ 3 =6

Note: 6 is the smallest perfect number.

Next perfect number is 28 since 1+ 2 + 4 + 7 + 14 = 28
Some more perfect numbers: 496, 8128

Code 1:

1. **C program to check perfect number**

```
#include<stdio.h>
int main(){
  int n,i=1,sum=0;

  printf("Enter a number: ");
  scanf("%d",&n);

  while(i<n){
      if(n%i==0)
          sum=sum+i;
        i++;
  }
  if(sum==n)
      printf("%d is a perfect number",i);
  else
      printf("%d is not a perfect number",i);

  return 0;
}
```

Code 2:

```c
#include<stdio.h>
int main(){
  int n,i,sum;
  int min,max;

  printf("Enter the minimum range: ");
  scanf("%d",&min);

  printf("Enter the maximum range: ");
  scanf("%d",&max);

  printf("Perfect numbers in given range is: ");
  for(n=min;n<=max;n++){
    i=1;
    sum = 0;

    while(i<n){
      if(n%i==0)
           sum=sum+i;
         i++;
    }

    if(sum==n)
      printf("%d ",n);
  }

  return 0;
}
```

Code 3:

3. **C program to print perfect numbers from 1 to 100**

```
#include<stdio.h>
int main(){
  int n,i,sum;

  printf("Perfect numbers are: ");
  for(n=1;n<=100;n++){
    i=1;
    sum = 0;

    while(i<n){
      if(n%i==0)
           sum=sum+i;
          i++;
    }

    if(sum==n)
      printf("%d ",n);
  }

  return 0;
}
```

Output:

```
Perfect numbers are: 6 28
```

****

ARMSTRONG NUMBER

Definition of Armstrong number or what is an Armstrong number:

Definition according to c programming point of view:

THOSE NUMBERS WHICH SUM OF THE CUBE OF ITS DIGITS IS EQUAL TO THAT NUMBER ARE KNOWN AS ARMSTRONG NUMBERS. FOR EXAMPLE 153 SINCE 1^3 + 5^3 + 3^3 = 1+ 125 + 9 =153

OTHER ARMSTRONG NUMBERS: 370,371,407 ETC.

IN GENERAL DEFINITION:

THOSE NUMBERS WHICH SUM OF ITS DIGITS TO POWER OF NUMBER OF ITS DIGITS IS EQUAL TO THAT NUMBER ARE KNOWN AS ARMSTRONG NUMBERS.

EXAMPLE 1: 153

TOTAL DIGITS IN 153 IS 3

AND 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153

Example 2: 1634

Total digits in 1634 is 4

And 1^4 + 6^4 + 3^4 +4^4 = 1 + 1296 + 81 + 64 =1634

Examples of Armstrong numbers: 1, 2, 3, 4, 5, 6, 7, 8, 9, 153, 370, 371, 407, 1634, 8208, 9474, 54748, 92727, 93084, 548834, 1741725

Code 1:

```c
#include<stdio.h>
int main(){
    int num,r,sum=0,temp;

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

    temp=num;
    while(num!=0){
        r=num%10;
        num=num/10;
        sum=sum+(r*r*r);
    }
    if(sum==temp)
        printf("%d is an Armstrong number",temp);
    else
        printf("%d is not an Armstrong number",temp);

    return 0;
}
```


The time complexity of a program that determines Armstrong number is: O (Number of digits)

Code 2:

```c
#include<stdio.h>
int main(){
    int num,r,sum,temp;
    int min,max;
    printf("Enter the minimum range: ");
    scanf("%d",&min);

    printf("Enter the maximum range: ");
    scanf("%d",&max);

    printf("Armstrong numbers in given range are: ");
    for(num=min;num<=max;num++){
        temp=num;
        sum = 0;
        while(temp!=0){
            r=temp%10;
            temp=temp/10;
            sum=sum+(r*r*r);
        }
        if(sum==num)
            printf("%d ",num);
    }

    return 0;
}
```

Code 3:

```c
#include<stdio.h>
int main(){
    int num,r,sum=0,temp;

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

    for(temp=num;num!=0;num=num/10){
        r=num%10;
        sum=sum+(r*r*r);
    }
    if(sum==temp)
        printf("%d is an Armstrong number",temp);
    else
        printf("%d is not an Armstrong number",temp);

    return 0;
}
```

Code 4:

```c
#include<stdio.h>
int main(){
    int num,r,sum,temp;

    for(num=1;num<=500;num++){
        temp=num;
        sum = 0;

        while(temp!=0){
            r=temp%10;
            temp=temp/10;
            sum=sum+(r*r*r);
        }
        if(sum==num)
            printf("%d ",num);
    }

    return 0;
}
```

****

PRIME NUMBER OR NOT

Definition of prime number:

A natural number greater than one has not any other divisors except 1 and itself. In other word we can say which has only two divisors 1 and number itself. For example: 5

Their divisors are 1 and 5.

Note: 2 is only even prime number.

Logic for prime number in c

We will take a loop and divide number from 2 to number/2. If the number is not divisible by any of the numbers then we will print it as prime number.

Example of prime numbers : 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199 etc.

Code 1:

```c
#include<stdio.h>

int main(){

    int num,i,count=0;
    printf("Enter a number: ");
    scanf("%d",&num);
    for(i=2;i<=num/2;i++){
        if(num%i==0){
         count++;
            break;
        }
    }
   if(count==0 && num!= 1)
        printf("%d is a prime number",num);
   else
      printf("%d is not a prime number",num);
   return 0;
}
```

```
Sample output:
Enter a number: 5
5 is a prime number
```

Code 2:

```c
#include<stdio.h>

int main(){
    int num,i,count;

    for(num = 1;num<=100;num++){
         count = 0;

         for(i=2;i<=num/2;i++){
             if(num%i==0){
                 count++;
                 break;
             }
        }

         if(count==0 && num!= 1)
             printf("%d ",num);
    }

   return 0;
}
```

Output:

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71
73 79 83 89 97
```

Code 3:

```c
#include<stdio.h>

int main(){

    int num,i,count,n;
    printf("Enter max range: ");
    scanf("%d",&n);

    for(num = 1;num<=n;num++){

         count = 0;

         for(i=2;i<=num/2;i++){
             if(num%i==0){
                 count++;
                 break;
             }
        }

         if(count==0 && num!= 1)
             printf("%d ",num);
    }

   return 0;
}
```

```
Sample output:
Enter max range: 50
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47
```

Code 4:

```c
#include<stdio.h>

int main(){

   int num,i,count,min,max;

    printf("Enter min range: ");
    scanf("%d",&min);

    printf("Enter max range: ");
    scanf("%d",&max);

    num = min;
    while(num<=max){

         count = 0;
         i=2;

         while(i<=num/2){
             if(num%i==0){
                 count++;
                 break;
             }
             i++;
        }

         if(count==0 && num!= 1)
             printf("%d ",num);

         num++;
    }

   return 0;
}
```

```
Sample output:
Enter min range: 50
Enter max range: 100
53 59 61 67 71 73 79 83 89 97
```

Code 5:

```c
#include<stdio.h>

int main(){

    int num,i,count,min,max;

     printf("Enter min range: ");
     scanf("%d",&min);

    printf("Enter max range: ");
    scanf("%d",&max);

    for(num = min;num<=max;num++){

         count = 0;

         for(i=2;i<=num/2;i++){
             if(num%i==0){
                 count++;
                 break;
             }
        }

         if(count==0 && num!= 1)
             printf("%d ",num);
    }

   return 0;
}
```

```
Sample output:
Enter min range: 10
Enter max range: 50
11 13 17 19 23 29 31 37 41 43 47
```

Code 6:

```c
#include<stdio.h>

int main(){

    int num,i,count,sum=0;

    for(num = 1;num<=100;num++){

         count = 0;

         for(i=2;i<=num/2;i++){
             if(num%i==0){
                 count++;
                 break;
             }
        }

         if(count==0 && num!= 1)
             sum = sum + num;
    }

    printf("Sum of prime numbers is: %d ",sum);

   return 0;
}
```

Output:
Sum of prime numbers is: 1060

Code 7:

```
#include<stdio.h>

int main(){

    int num,i,count,min,max,sum=0;

     printf("Enter min range: ");
     scanf("%d",&min);

    printf("Enter max range: ");
    scanf("%d",&max);

    for(num = min;num<=max;num++){

         count = 0;

         for(i=2;i<=num/2;i++){
             if(num%i==0){
                 count++;
                 break;
             }
        }

         if(count==0 && num!= 1)
             sum = sum + num;
    }

    printf("Sum of prime numbers is: %d ",sum);

   return 0;
}
```

****

STRONG NUMBER OR NOT

Definition of strong number:

A number is called strong number if sum of the factorial of its digit is equal to number itself. For example: 145 since
1! + 4! + 5! = 1 + 24 + 120 = 145

Code 1:

```
#include<stdio.h>
int main(){
  int num,i,f,r,sum=0,temp;

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

  temp=num;
  while(num){
      i=1,f=1;
      r=num%10;
      while(i<=r){
        f=f*i;
        i++;
      }
      sum=sum+f;
      num=num/10;
  }
  if(sum==temp)
      printf("%d is a strong number",temp);
  else
      printf("%d is not a strong number",temp);

  return 0;
}
```

Code 2:

```
#include<stdio.h>
int main(){
  int num,i,f,r,sum,temp;
  int min,max;

  printf("Enter minimum range: ");
  scanf("%d",&min);

  printf("Enter maximum range: ");
  scanf("%d",&max);

  printf("Strong numbers in given range are: ");
  for(num=min; num <= max; num++){
      temp = num;
      sum=0;

      while(temp){
           i=1;
           f=1;
           r=temp%10;

           while(i<=r){
             f=f*i;
             i++;
           }
         sum=sum+f;
         temp=temp/10;
      }

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

```
Sample output:
Enter minimum range: 100
Enter maximum range: 100000
Strong numbers in given range are: 145 40585
```

****

EVEN OR ODD NUMBER

Number is called even number if it is divisible by two otherwise odd.

Example of even numbers: 0,2,4,8,9,10 etc.
Example of odd numbers: 1, 3,5,7,9 etc.

Code 1:

```
#include<stdio.h>

int main(){

    int number;

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

    if(number % 2 ==0)
        printf("%d is even number.",number);
    else
        printf("%d is odd number.",number);

    return 0;

}
```

Sample output:
Enter any integer: 5
5 is odd number.

Code 2:

```c
#include<stdio.h>

int main(){

    int number;
    int min,max;

    printf("Enter the minimum range: ");
    scanf("%d",&min);

    printf("Enter the maximum range: ");
    scanf("%d",&max);

    printf("Odd numbers in given range are: ");
    for(number = min;number <= max; number++)

        if(number % 2 !=0)
            printf("%d ",number);

    return 0;

}
```

```
Sample output:
Enter the minimum range: 1
Enter the maximum range: 20
Odd numbers in given ranges are: 1 3 5 7 9 11 13 15 17
19
```

Code 3:

```c
#include<stdio.h>

int main(){

    int number;
    int min,max;

    printf("Enter the minimum range: ");
    scanf("%d",&min);

    printf("Enter the maximum range: ");
    scanf("%d",&max);

    printf("Odd numbers in given range are: ");
    for(number = min;number <= max; number++)

        if(number % 2 !=0)
            printf("%d ",number);

    printf("\nEven numbers in given range are: ");
    for(number = min;number <= max; number++)

        if(number % 2 ==0)
            printf("%d ",number);

    return 0;
}
```

Sample output:

```
Enter the minimum range: 1
Enter the maximum range: 20
Odd numbers in given ranges are: 1 3 5 7 9 11 13 15 17
19
Even numbers in given ranges are: 2 4 6 8 10 12 14 16
18 20
```

Code 4:

```
#include<stdio.h>

int main(){

    int number;
    int min,max;
    long sum =0;

    printf("Enter the minimum range: ");
    scanf("%d",&min);

    printf("Enter the maximum range: ");
    scanf("%d",&max);

    for(number = min;number <= max; number++)
        if(number % 2 !=0)
            sum = sum + number;

    printf("Sum of odd numbers in given range is:
%ld",sum);

    return 0;

}
```

```
Sample output:
Enter the minimum range: 1
Enter the maximum range: 100
Sum of odd numbers in given range is: 2500
```

Code 5:

```
#include<stdio.h>

int main(){

    int number;
    int min,max;
    long odd_sum =0,even_sum = 0;

    printf("Enter the minimum range: ");
    scanf("%d",&min);

    printf("Enter the maximum range: ");
    scanf("%d",&max);

    for(number = min;number <= max; number++)
        if(number % 2 != 0)
            odd_sum = odd_sum + number;
        else
            even_sum = even_sum + number;

    printf("Sum of even numbers in given range is:
%ld\n",even_sum);
    printf("Sum of odd numbers in given range is:
%ld",odd_sum);

    return 0;

}
```

```
Sample output:
Enter the minimum range: 1
Enter the maximum range: 10
Sum of even numbers in given range is: 30
Sum of odd numbers in given range is: 25
```

****

PALINDROME OR NOT (NUMBER)

Definition of Palindrome number or What is palindrome number?

A number is called palindrome number if it is remain same when its digits are reversed. For example 121 is palindrome number. When we will reverse its digit it will remain same number i.e. 121

Palindrome numbers examples: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 22, 33, 44, 55, 66, 77, 88, 99, 101, 111, 121, 131, 141, 151, 161, 171, 181, 191 etc.

Code 1:

```
#include<stdio.h>
int main(){
    int num,r,sum=0,temp;

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

    temp=num;
    while(num){
        r=num%10;
        num=num/10;
        sum=sum*10+r;
    }
    if(temp==sum)
        printf("%d is a palindrome",temp);
    else
        printf("%d is not a palindrome",temp);

    return 0;
}
```

Sample output:

```
Enter a number: 131
131 is a palindrome
```

Code 2:

```c
#include<stdio.h>
int main(){
    int num,r,sum,temp;
    int min,max;

    printf("Enter the minimum range: ");
    scanf("%d",&min);

    printf("Enter the maximum range: ");
    scanf("%d",&max);

    printf("Palindrome numbers in given range are: ");
    for(num=min;num<=max;num++){
        temp=num;
        sum=0;

        while(temp){
            r=temp%10;
            temp=temp/10;
            sum=sum*10+r;
        }
        if(num==sum)
            printf("%d ",num);
    }
    return 0;
}
```

```
Sample output:
Enter the minimum range: 1
Enter the maximum range: 50
Palindrome numbers in given range are: 1 2 3 4 5 6 7 8
9 11 22 33 44
```

Code 3:

```c
#include<stdio.h>
int main(){
    int num,r,sum=0,temp;

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

    for(temp=num;num!=0;num=num/10){
        r=num%10;
        sum=sum*10+r;
    }
    if(temp==sum)
        printf("%d is a palindrome",temp);
    else
        printf("%d is not a palindrome",temp);

    return 0;
}
```

```
Sample output:
Enter a number: 1221
1221 is a palindrome
```

Code 4:

```c
#include<stdio.h>

int checkPalindrome(int);
int main(){
    int num,sum;

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

    sum = checkPalindrome(num);

    if(num==sum)
        printf("%d is a palindrome",num);
    else
    printf("%d is not a palindrome",num);

    return 0;
}

int checkPalindrome(int num){

    static int sum=0,r;

    if(num!=0){
        r=num%10;
        sum=sum*10+r;
        checkPalindrome(num/10);
    }

    return sum;
}
```

Sample output:

```
Enter a number: 25
25 is not a palindrome
```

****

PALINDROME OR NOT (NUMBER)

Definition of Palindrome string:

A string is called palindrome if it symmetric. In other word a string is called palindrome if string remains same if its characters are reversed. For example: asdsa
If we will reverse it will remain same i.e. asdsa

Code 1

```
#include<string.h>
#include<stdio.h>
int main(){
  char *str,*rev;
  int i,j;
  printf("\nEnter a string:");
  scanf("%s",str);
  for(i=strlen(str)-1,j=0;i>=0;i--,j++)
      rev[j]=str[i];
      rev[j]='\0';
  if(strcmp(rev,str))
      printf("\nThe string is not a palindrome");
  else
      printf("\nThe string is a palindrome");
  return 0;
}
```

****

ROOTS OF QUADRATIC EQUATION

// TO BE WRITTEN

Code 1

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

int main(){
  float a,b,c;
```

```
  float d,root1,root2;


  printf("Enter a, b and c of quadratic equation: ");
  scanf("%f%f%f",&a,&b,&c);

  d = b * b - 4 * a * c;

  if(d < 0){
    printf("Roots are complex number.\n");

    printf("Roots of quadratic equation are: ");
    printf("%.3f%+.3fi",-b/(2*a),sqrt(-d)/(2*a));
    printf(", %.3f%+.3fi",-b/(2*a),-sqrt(-d)/(2*a));

    return 0;
  }
  else if(d==0){
   printf("Both roots are equal.\n");

   root1 = -b /(2* a);
   printf("Root of quadratic equation is: %.3f
",root1);

   return 0;
  }
  else{
   printf("Roots are real numbers.\n");

   root1 = ( -b + sqrt(d)) / (2* a);
   root2 = ( -b - sqrt(d)) / (2* a);
   printf("Roots of quadratic equation are: %.3f ,
%.3f",root1,root2);
  }

  return 0;
}
```

```
Sample output:
Enter a, b and c of quadratic equation: 2 4 1
Roots are real numbers.
Roots of quadratic equation are: -0.293, -1.707
```

Code 2

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

int main(){
  float a,b,c;
  float d,root1,root2;

  printf("Enter quadratic equation in the format
ax^2+bx+c: ");
  scanf("%fx^2%fx%f",&a,&b,&c);

  d = b * b - 4 * a * c;

  if(d < 0){
    printf("Roots are complex number.\n");

    return 0;
  }

   root1 = ( -b + sqrt(d)) / (2* a);
   root2 = ( -b - sqrt(d)) / (2* a);
   printf("Roots of quadratic equation are: %.3f ,
%.3f",root1,root2);

  return 0;
}
```

Sample output:

```
Enter quadratic equation in the format ax^2+bx+c:
2x^2+4x+-1
Roots of quadratic equation are: 0.000, -2.000
```

****

FIBONACCI SERIES

Definition of Fibonacci numbers:

We assume first two Fibonacci are 0 and 1
A series of numbers in which each sequent number is sum of its two previous numbers is known as Fibonacci series and each numbers are called Fibonacci numbers. So Fibonacci numbers is

Algorithm for Fibonacci series

$F_n = F_{n-2} + F_{n-1}$

Code 1:

```
#include<stdio.h>
int main(){
    int k,r;
    long int i=01,j=1,f;

    //Taking maximum numbers form user
    printf("Enter the number range:");
    scanf("%d",&r);

    printf("FIBONACCI SERIES: ");
    printf("%ld %ld",i,j); //printing firts two values.

    for(k=2;k<r;k++){
        f=i+j;
        i=j;
        j=f;
        printf(" %ld",j);
    }

    return 0;
}
```

Code 2:

1. **Fibonacci series using array in c**
2. **Fibonacci series program in c language**
3. **Source code of Fibonacci series in c**
4. **Wap to print Fibonacci series in c**

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

    int i,range;
    long int arr[40];

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

    arr[0]=0;
    arr[1]=1;

    for(i=2;i<range;i++){
        arr[i] = arr[i-1] + arr[i-2];
    }

    printf("Fibonacci series is: ");
    for(i=0;i<range;i++)
        printf("%ld ",arr[i]);

    return 0;
}
```

```
Sample output:
Enter the number range: 20
Fibonacci series is: 0 1 1 2 3 5 8 13 21 34 55 89 144
233 377 610 987 1597 2584 4181
```

Code 3:

```c
#include<stdio.h>
int main(){
    int k=2,r;
    long int i=0l,j=1,f;

    printf("Enter the number range:");
    scanf("%d",&r);

    printf("Fibonacci series is: %ld %ld",i,j);

    while(k<r){
         f=i+j;
         i=j;
         j=f;
         printf(" %ld",j);
          k++;
    }

    return 0;
}
```

```
Sample output:
Enter the number range: 10
Fibonacci series is: 0 1 1 2 3 5 8 13 21 34
```

Code 4:

```
#include<stdio.h>
int main(){
    int k,r;
    long int i=0,j=1,f;
    long int sum = 1;

    printf("Enter the number range: ");
    scanf("%d",&r);

    for(k=2;k<r;k++){
        f=i+j;
        i=j;
        j=f;
        sum = sum + j;
    }

    printf("Sum of Fibonacci series is: %ld",sum);

    return 0;
}
```

Sample output:

```
Enter the number range: 4
Sum of Fibonacci series is: 4
```

****

FACTORIAL OF A NUMBER

Definition / Algoritham

```
Factorial of number is defined as:
Factorial (n) = 1*2*3 … * n
For example: Factorial of 5 = 1*2*3*4*5 = 120
Note: Factorial of zero = 1
```

Code 1:

```
#include<stdio.h>
int main(){
  int i=1,f=1,num;

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

  while(i<=num){
      f=f*i;
      i++;
  }

  printf("Factorial of %d is: %d",num,f);
  return 0;
}
```

```
Sample output:
Enter a number: 5
Factorial of 5 is: 120
```

Code 2:

```c
#include<stdio.h>
int main(){
  int i,f=1,num;

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

  for(i=1;i<=num;i++)
      f=f*i;

  printf("Factorial of %d is: %d",num,f);
  return 0;
}
```

Code 3:

```c
#include<stdio.h>

void findFactorial(int,int *);
int main(){
  int i,factorial,num;

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

  findFactorial(num,&factorial);
  printf("Factorial of %d is: %d",num,*factorial);

  return 0;
}

void findFactorial(int num,int *factorial){
    int i;

    *factorial =1;

    for(i=1;i<=num;i++)
      *factorial=*factorial*i;
}
```

Code 4:

1. **Factorial program in c using function**
2. **C program to find factorial of a number**

```
#include<stdio.h>

int findFactorial(int);
int main(){
  int i,factorial,num;

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

  factorial = findFactorial(num);
  printf("Factorial of %d is: %d",num,factorial);

  return 0;
}

int findFactorial(int num){
    int i,f=1;

    for(i=1;i<=num;i++)
      f=f*i;

     return f;
}
```

Sample output:

```
Enter a number: 8
Factorial of 8 is: 40320
```

Code 5:

1. **Factorial series in c**

```c
#include<stdio.h>
int main(){
  long f=1;
  int i,num,min,max;

  printf("Enter the minimum range: ");
  scanf("%d",&min);

  printf("Enter the maximum range: ");
  scanf("%d",&max);

  printf("Factorial series in given range: ");
  for(num=min;num<=max;num++){
    f=1;

    for(i=1;i<=num;i++)
      f=f*i;

    printf("%ld ",f);
  }

  return 0;
}
```

```
Sample output:
Enter the minimum range: 1
Enter the maximum range: 10
Factorial series in given range: 1 2 6 24 120 720 5040
40320 362880 3628800
```

****

FLOYD'S TRIANGLE

Definition of floyd's triangle:

Floyd's triangle is a right angled-triangle using the natural numbers. Examples of floyd's triangle:

Example 1:

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

Example 2:

```
1
2   3
4   5   6
7   8   9   10
11  12  13  14  15
16  17  18  19  20 21
```

1. Write a c program to print Floyd's triangle

2. C program to display Floyd's triangle

3. How to print Floyd's triangle in c

```
#include<stdio.h>


int main(){

  int i,j,r,k=1;

  printf("Enter the range: ");
  scanf("%d",&r);

  printf("FLOYD'S TRIANGLE\n\n");
  for(i=1;i<=r;i++){
      for(j=1;j<=i;j++,k++)
           printf(" %d",k);
      printf("\n");
  }

  return 0;
}
```

```
Sample output:
Enter the range: 10
FLOYD'S TRIANGLE

 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
```

****

PASCAL'S TRIANGLE

```
Enter the no. of lines: 8
       1
      1 1
     1 2 1
    1 3 3 1
   1 4 6 4 1
  1 5 10 10 5 1
 1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
```

```c
#include<stdio.h>

long fact(int);
int main(){
    int line,i,j;

    printf("Enter the no. of lines: ");
    scanf("%d",&line);

    for(i=0;i<line;i++){
         for(j=0;j<line-i-1;j++)
             printf(" ");

         for(j=0;j<=i;j++)
             printf("%ld ",fact(i)/(fact(j)*fact(i-
j)));
         printf("\n");
    }
    return 0;
}

long fact(int num){
    long f=1;
    int i=1;
    while(i<=num){
         f=f*i;
         i++;
  }
  return f;
 }
```

Sample output:

```
Enter the no. of lines: 8
       1
      1 1
     1 2 1
    1 3 3 1
   1 4 6 4 1
  1 5 10 10 5 1
 1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
```

****

MULTIPLICATION TABLE

// WAITLISTED

```
#include<stdio.h>
int main(){
  int r,i,j,k;
  printf("Enter the number range: ");
  scanf("%d",&r);
  for(i=1;i<=r;i++){
      for(j=1;j<=10;j++)
          printf("%d*%d=%d ",i,j,i*j);
      printf("\n");
  }
  return 0;
}
```

Sample Output:

```
Enter the number range: 5

1*1=1 1*2=2 1*3=3 1*4=4 1*5=5 1*6=6 1*7=7 1*8=8 1*9=9 1*10=10

2*1=2 2*2=4 2*3=6 2*4=8 2*5=10 2*6=12 2*7=14 2*8=16 2*9=18 2*10=20
3*1=3 3*2=6 3*3=9 3*4=12 3*5=15 3*6=18 3*7=21 3*8=24 3*9=27 3*10=30
4*1=4  4*2=8  4*3=12  4*4=16  4*5=20  4*6=24  4*7=28  4*8=32  4*9=36
4*10=40
5*1=5  5*2=10  5*3=15  5*4=20  5*5=25  5*6=30  5*7=35  5*8=40  5*9=45
5*10=50
```

****

PRINTING ASCII VALUES' USING C-PROGRAM

// PRINT'S ASCII VALUE OF ALL 255 WORD'S OR CHAR OR SPL.SYMBOLS

```
#include<stdio.h>


int main(){


    int i;

    for(i=0;i<=255;i++)
         printf("ASCII value of character %c:
%d\n",i,i);

    return 0;
}
```

****

print hello world without using semicolon

LOGICS TO PRINT HELLO WORLD

```
#include<stdio.h>
void main(){
    if(printf("Hello world")){
    }
}
```

Solution: 2

```
#include<stdio.h>
void main(){
    while(!printf("Hello world")){
    }
}
```

Solution: 3

```
#include<stdio.h>
void main(){
    switch(printf("Hello world")){
    }
}
```

****
own source code as its output

```
#include<stdio.h>

int main(){
    FILE *fp;
    char c;

    fp = fopen(__FILE__,"r");

    do{
        c= getc(fp);
        putchar(c);
    }
    while(c!=EOF);

    fclose(fp);

    return 0;
}

Output:
#include<stdio.h>
int main(){
    FILE *fp;
    char c;

    fp = fopen(__FILE__,"r");

    do{
        c= getc(fp);
        putchar(c);
    }
    while(c!=EOF);

    fclose(fp);

    return 0;
}
```

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****