PrepInsta Handbook for Top 100 Codes

For Placement Preparation

PrepInsta Technologies Pvt Ltd

Preface:

This book contains all the information regarding Top 100 codes which is asked in Placement Tests and Interview Rounds. Nowadays you will see many books and online pages providing information on Coding questions. Mostly those books contain one section i.e either the coding questions or the theoretical part. There is no such proper book providing all the updated information at one single place.

This book contains various questions and theory knowledge of all the types asked in Placement tests. This book carries all the Top 100 codes asked during the whole Recruitment Process. .

It is hoped that the subject matter will instill trust in the applicants, and that the book will assist them in finding an ideal teacher.

Disclaimer: This book is made and published under the complete knowledge and expertise of the Author, however if there will be any error then it will be taken care of in the next Revised edition. Constructive suggestions and feedback are most welcome by our esteemed readers.

First Edition

May 2021
Edition: 1 (2021)

Published on GSM Paper with PrepInsta WaterMark

Publisher Address: K-3, Director Vidyamandir Shastri Nagar, Meerut, UP, 250004

Type settings : PrepInsta Technologies Private Limited

Printer Name: Creative Print Arts

Price- Rs.999/-

- The correct price of the book is already mentioned on this page. Any revised price on the frontend or backend of the book is not acceptable.

Address of the Publisher:

K-3, Director Vidyamandir Shastri Nagar, Meerut, UP, 250004

Publication Team:

PrepInsta Technologies Private Limited

Contents

Chapter 1. Positive or Negative number

The following concept will test whether a number is positive or negative. It is done by checking where the number lies on the number line. The following algorithm will help to check this condition.

- If the input number is greater than zero then it is a positive number.
- If the input number is less than zero it is a negative number.
- If the number is zero then it is neither positive nor negative.

Same logic we have followed in the below C program.**Working**

- Step 1. Start
- Step 2. Enter the number.
- Step 3. If the number is less than or equal to zero, check if it is zero.
- Step 4. If the number is zero, print, “, The number is zero.”
- Step 5. If the number is less than zero, print, “The number is negative.”
- Step 6. If the number is more than zero, print, “The number is positive.”
- Step 7. Stop

C Code :

```
#include<stdio.h>
  int main()
  {
  int num;
  printf(“Insert a number: “);
  scanf(“%d”, &num);
  //Condition to check if the number is negative or
positive
  if (num <= 0)
  {
  if (num == 0)
    printf(“The number is 0.”);
  else
    printf(“The number is negative”);
  }
  else
    printf(“The number is positive”);
  return 0;
  }
```

C++ Code :

```
#include<iostream>
using namespace std;
int main()
{
  #ifndef ONLINE_JUDGE
    // for getting input from input.txt
    freopen(“input1.txt”, “r”, stdin);
    // for writing output to output.txt
    freopen(“output.txt”, “w”, stdout);
    #endif
  int no;2
  cout<<“Enter a number:”;
  cin>>no;
  if(no==0)
  {
    cout<<“0 is neither positive nor negative”;
  }
  else if(no>0)
  {
    cout<<no<<“is a positive number”;
  }
  else
  {
     cout<<no<<“is a negative number”;
  }
  return 0;
}
```

Java Code :

```
//Java program to check a number is positive or negative
import java.util.Scanner;
public class pos_or_neg
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from the user
                System.out.print("Enter a Number : ");
                int numb = sc.nextInt();
                //condition for positive
                if(numb > 0)

System.out.println("Positive");
                //condition for negative
                else if(numb < 0)

System.out.println("Negative");
                else
                        System.out.println("Zero");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
num = int(input("Insert a number:"))
if num > 0:
   print("The number is Positive")
else:
   print("The number is Negative")
```

Chapter 2. Even or Odd number

We can determine whether a number is even or odd. This can be tested using different methods. The test can be done using simple methods such as testing the number's divisibility by 2. If the remainder is zero, the number is even. If the remainder is not zero, then the number is odd. The following algorithm describes how a C program can test if a number is even or odd.

Example :

Number is 24

It is an even number because it is exactly divisible by 2

Number is 15

It is odd number because it is not divisible by 2

Working

- Step 1. Start
- Step 2. Enter a number.
- Step 3. If the number is divisible by 2, it is even.
- Step 4. If the number is not divisible by 2, it is odd.
- Step 5. Stop

C Code :

```
#include<stdio.h>
int main()
{
    int number;
    printf(“Insert a number \n“);
    scanf(“%d”,&number);

  //Checking if the number is divisible by 2
   if (number%2 == 0)
     printf(“The number is even\n“);

   else
     printf(“The number is odd\n“);
  return 0;
 }
```

C++ Code :

```
//C++ Program
// number is even or odd
#include
using namespace std;
int main()
{
   cout<<“Enter a number: “;
   int check;
   cin>>check;
   //checking whether the number is even or odd
   if(check % 2 == 0)
   {
      cout<<check<<” is an even number”;
   }
   else
   {
      cout<<check<<” is an odd number”;
   }
   return 0;
}
```

Java Code :

```
//Java Program to check a number is even or odd
import java.util.Scanner;
public class even_or_odd
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from the user
                System.out.print("Enter a Number : ");

                int numb = sc.nextInt();
                //condition for even
                if(numb  %  2 == 0)

                        System.out.println("Even
Number");
                //condition for odd
                else


                        System.out.println("Odd
Number");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();

        }
}
```

Python Code :

```
num = int(input("Enter a Number:"))
if num % 2 == 0:
   print("Given number is Even")
else:
   print("Given number is Odd")

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 3. Sum of First N Natural numbers

A Natural number is the same as Counting number.They are used to Count the numbers of real physical objects. Natural numbers start from 1 and go on infinite. The positive numbers 1, 2, 3… are known as natural numbers.
Example:
Natural number={1,2,,4,5,6,…….}.
Formula for Sum of First N natural numbers is : n(n+1)/2.
If you want to add the first 5 Natural numbers then we find the Sum of 1+2+3+4+5 =15.

Working

Step 1. Start

Step 2. Enter a number (N).

Step 3. Use formula to calculate the sum of N natural

number || Sum=(n*(n+1))/2.

Step 4. Print sum of N Natural Number.

Step 5. Stop

C Code :

```
 #include<stdio.h>
 int main()
 {
  int sum = 0, n;
  printf("Enter the first N Natural Number\n");
  scanf("%d",&n);
  sum=(n*(n+1))/2;
  printf("sum is %d",sum);
  return 0;
 }
```

C++ Code :

```
import java.util.*;
class prepinsta
{
    public static void main(String[] aa){
    Scanner sc=new Scanner(System.in);
  int sum=0;
System.out.println("Enter the vlue of n");
   int n=sc.nextInt();
   for(int i=1;i<=n;i++)
   sum=sum+i;
System.out.println("Sum is " +sum);

}
}
```

Java Code :

```
import java.util.*;
class prepinsta
{
    public static void main(String[] aa){
    Scanner sc=new Scanner(System.in);
  int sum=0;
System.out.println("Enter the value of n");
   int n=sc.nextInt();
   for(int i=1;i<=n;i++)
   sum=sum+i;
System.out.println("Sum is " +sum);

}
}
```

Python Code :

Method 1:

```
num = int(input("Enter the Number:"))
value = 0
for i in range(1, num+1):
    value = value + i

print("Sum of N natural numbers:", value)
```

Method 2:

```
num = int(input("Enter the Number:"))
sum = (num * (num+1))/2
print("The Sum of N natural Number is {}".format(sum))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 4. Sum Of N Natural Numbers

In the C programming language, the user is allowed to insert any integer value. With the help of For loop, this C program can calculate the sum of N natural numbers. Within this program, the first printf statement will request the user to insert a number or value then the scanf statement will allocate the user inserted value to integer variable. The sum is calculated in the For loop.

To perform the arithmetic operation of addition of n numbers we use this conditions
Example –
Enter Number 3
N natural numbers 1,2,3,4,5,6,7,8…….
Where first 3 number is 1,2,3
Then we will return sum of number = 6
Working

- Step 1. Start
- Step 2. Enter a number (N).
- Step 3. Use "For loop" to iterate upto the user inserted value.
- Step 4. The "For loop" will calculate the sum of the user inserted value.
- Step 5. Stop

C Code :

```
/* C Program to find Sum of N Numbers using For Loop */
   #include<stdio.h>
   int main()
   {
   //for initialize variable
   int Number, i, Sum = 0;
   //to take user input
   printf ("\n Kindly Insert an Integer Variable\n");
   scanf ("%d", &Number);

   //use for loop for these condition
   for(i = 1; i <= Number; i++)
   {
   Sum = Sum + i;
   }
   //display
   printf ("Sum of Natural Numbers = %d", Sum);

   return 0;
   }
```

Output

```
Kindly insert an integer variable : 5
Sum of Natural Numbers = 15
```

C++ Code :

```
//C++ Program
   // Sum of n natural numbers
   #include<iostream>
   using namespace std;
   int main()
   {

     int sum , N;
      cout << "Enter the limit: ";
      //user input
      cin >> N;
      //calculating
      sum sum= N*(N+1)/2;
      cout<<"The Sum of first "<< N <<" Natural Numbers
is "<< sum;
      return 0;
   }
```

Java Code :

```
//Java program to print the sum of n natural numbers
import java.util.Scanner;
public class sum_of_n_natural_numbers
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int n = sc.nextInt();
                //declare a variable to store sum
                int sum=0;
                //loop to add n natural numbers
```

```
                        for(int i = 1 ; i <= n ; i++)
                        sum=sum+i;
                        //display the sum
                        System.out.print("Sum of n natural
numbers is "+sum);
                        //closing scanner class(not compulsory,
but good practice)
                        sc.close();


            }
}
```

Python Code :

Method 1:

```
num = int(input("Enter the Number:"))
value = 0
for i in range(1, num+1):
    value = value + i

print("Sum of N natural numbers:", value)
```

Method 2:

```
num = int(input("Enter the Number:"))
sum = (num * (num+1))/2
print("The Sum of N natural Number is {}".format(sum))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 5. Sum of numbers in a given range

The program given below accepts a range of values and calculates their sum. The program uses a loop to calculate the sum of the values provided by the user. The following section presents an algorithm followed by a C program to calculate this sum.

Example:-Enter first and last range 4 and 8.

To use for loops start at 4 and end 8 and sum off inside the no.in this range.

Answer is 30(i.e 4+5+6+7+8=30).

Problem Description

In this Program to find the sum of numbers in a given range. In the C program we are using a for loop. In that loop we have to start a first range given by the user and last range also input by the user. And perform the arithmetic operation of sum of number

The program given below accepts a range of values and calculates their sum. The program uses a loop to calculate the sum of the values provided by the user. Also, the provided numbers must be in integer format for successful calculation. The following section presents an algorithm followed by a C program to calculate this sum.

Example:-Enter first and last range 4 and 8.

To use for loop start at 4 and end 8 and sum off inside the no.in this range.

Answer is 30(i.e 4+5+6+7+8=30).

Working

- Step 1. Initialize variables (firstrange,lastrange, total and i).
- Step 2. Input fistrange and lastrange by user.
- Step 3. We use “for loop” with the condition (i=firstrange;i<= lastrange;i++).When loop will work until i= secondrange.
- Step 4. The loop will start with i=firstrange and end with i<= lastrange.
- Step 6. In the loop for every cycle total will be incremented by i.
- Step 7. Then condition false print sum of number(total).
- Step 8. Stop

C Code :

```
 #include <stdio.h>
   int main()
   {
   //for initialization of variable
   int firstrange,lastrange, i=0, total= 0;

   //to use user input first range & last range
   printf(“Enter the value first range and last range\n“);
   scanf(“%d\n%d”,&firstrange, &lastrange);

   //use for loop for total no.inside the range
   for(i = firstrange; i <= lastrange; i++){

      //total+=i;
      total = total + i;
   }

   //print the sum of number
   printf(“Sum of number firstrang  %d to lastrange %d is:
%d”,firstrange, lastrange, total);
   }
   Output

   Enter the value first range and last range:
   30
   40
   sum of number firstrange 30 to lastrange 40 is :  385
```

C++ Code :

```
//C++ Program
   //Sum of Natural Numbers in a given range
   #include<iostream>
   using namespace std;
   //main Program
   int main()
   {
      int sum = 0 , upper_limit, lower_limit;
      cout << “Enter the lower limit: “;
      cin >> lower_limit;


      cout << “Enter the upper limit: “;
      cin >> upper_limit;

      //calculating sum of numbers in the given range
      for(int i = lower_limit; i <= upper_limit; i++){
      sum += i;
      }

      //printing output
      cout<<“The Sum of Natural Numbers from “ <<
lower_limit << ” to “ << upper_limit << ” is “ << sum;
      return 0;
   }
```

Java Code :

```
//Java program to print the sum of numbers in a given range
import java.util.Scanner;
public class sum_of_numbers_in_range
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input from user
                        System.out.print("Enter starting
number : ");
                        int start = sc.nextInt();
                        System.out.print("Enter ending number
: ");
                        int end = sc.nextInt();
                        //declare a variable to store sum
                        int sum = 0;
                        //loop to add n natural numbers
                        for(int i = start ; i <= end ; i++)
                        sum=sum+i;
                        //display the sum
                        System.out.print("Sum of numbers in
the range from "+start+" to "+end+" is "+sum);
                        //closing scanner class(not compulsory,
but good practice)
                        sc.close();


            }
}
```

Python Code :

Method 1:

```
num = int(input("Enter the Number:"))
value = 0
for i in range(1, num+1):
    value = value + i

print("Sum of N natural numbers:", value)
```

Method 2:

```
num = int(input("Enter the Number:"))
sum = (num * (num+1))/2
print("The Sum of N natural Number is {}".format(sum))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 6. Greatest of two numbers

In C programming language, the greatest of numbers can be identified with the help of IF-ELSE statements. The user is asked to insert two integers. The numbers inserted are then calculated using a set of programs to get the correct output. It will find the highest number among them using IF-ELSE Statement and start checking which one is larger to display the largest number.

Example – If the given numbers are 12 and 9 then greater number is 12

12, 9= 12>9

Working

Step 1: Start

Step 2: Insert two integers no1 and no2 from the user with the help of scanf statement.

Step 3: Check if the no1 is bigger in value than no2 using the if statement.

Step4: If no1 is greater, then print no1 using the printf statement, if the case is vice versa then check whether no2 is greater than no1 with the help of elseif statement.

Step 5: If no2 is greater than no1, then print no2 using printf statement, if not then print no1 and no2 are equal using printf statement.

Step 6: Stop

C Code:

```
#include<stdio.h>
  int main()
  {
  int no1, no2;
  printf("Insert two numbers:");
  scanf("%d %d",&no1, &no2);

  //Condition to check which of the two number is greater
  //it will compare of number where number 1 is greater
  if(no1 > no2)
     printf("%d is greatest",no1);

  //where number 2 is greater
  else if(no2 > no1)
     printf("%d is greatest",no2);

  //for both are equal
  else
     printf("%d and %d are equal", no1, no2);

  return 0;
  }
```

Output

```
Insert Two Numbers : 5
6
6 is the Greatest
```

C++ Code :

```
//C++ program
  //Greatest of two numbers
  #include<iostream>
  using namespace std;
  //main program
  int main()
  {
     int first,second;
     cout<<"Enter first number: ";
     cin>>first;
     cout<<"Enter second number: ";
     cin>>second;
     if(first==second)
     {
        cout<<"both are equal";
     }
     else if(first>second)
     {
        cout<<first<<" is greater than "<<second;
     }
     else
     {
        cout<<second<<" is greater than "<<first;
     }
     return 0;
  }
```

Java Code :

```
//Java program to find greatest of two numbers
import java.util.Scanner;
public class greatest_of_two_numbers
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input first number
                        System.out.print("Enter the first
number : ");
                        int first = sc.nextInt();
                        //input second number
                        System.out.print("Enter the second
number : ");
                        int second = sc.nextInt();
                        //conditions
                        if(first > second)

                                    System.out.println(first+" is
greater than "+second);
                        else if(second > first)
                                    System.out.println(second+"
is greater than "+first);
                        else
                                    System.out.println("Both
numbers are Equal");
                        //closing scanner class(not compulsory,
but good practice)
                        sc.close();


            }
}
```

Python Code :

```
first = int(input("Enter first number:"))
second = int(input("Enter second number:"))
if first > second:
   print("First is Greater than Second")
else:
   print("Second is Greater than First")
```

Chapter 7. Greatest of the Three numbers

The C program to find the greatest of three numbers requires the user to insert three integers. Flow chart is also used in C programming to find the greatest number among three integers inserted by the user. A simple if-else block is used to identify the greatest number.

Problem Description

C programs to find the greatest of three numbers require the user to insert three integers. Flow chart is also used in C programming to find the greatest number among three integers inserted by the user. A simple if-else block is used to identify the greatest number. The program will ask the user to insert three integer variables. And on the basis of the inserted number, the program will equate and exhibit the greatest number as an output. This program uses no1, no2 & no3 as three integer variables that are represented number1, number2 and number3 respectively in the program.

Working

Step 1: Start

Step 2: Take three integer values from the user.

Step 3: If no1 is greater than no2 and no3, printf "Number1 is greatest".

Step 4: If no2 is greater than no1 and no3, printf "Number2 is greatest".

Step 5: If both the conditions are false, then printf "Number3 is greatest".

Step 6: Stop

C code :

```
#include<stdio.h>
  int main()
  {
  int no1,no2,no3;

  //Prompt user to insert any three integer variables
  printf("\nInsert value of no1, no2 and no3:");
  scanf("%d %d %d", &no1, &no2, &no3);

  //for check of number 1 is greater
  if((no1 > no2) && (no1 > no3))
    printf("\n Number1 is greatest");

  //weather number 2 is grater
  else if((no2 > no3) && (no2 > no1))
    printf("\n Number2 is greatest");

  //other conditions are false than number 3 is greater
  else
    printf("\n Number3 is greatest");
  return 0;
  }
```

Output

```
Insert Value of No1, No2 and No3: 15, 200, 101
Number 2 is Greatest
```

C++ code :

```
//C++ Program
  //Greatest of three numbers
  #include<iostream>
  using namespace std;
  //main program
  int main()
  {
    int first, second, third;
    cout<<"Enter first number: ";
    cin>>first;
    cout<<"Enter second number: ";
    cin>>second;
    cout<<"Enter third number: ";
    cin>>third;
    //comparing first with other numbers
    if((first >= second) && (first >= third))
    {
      cout<<first<<" is the greatest";
    }
    //comparing Second with other numbers
    else if((second >= first) && (second >= third))
    {
      cout<<second<<" is the greatest";
    }
    else
    {
      cout<<third<<" is the greatest";
    }
    return 0;
  }
```

Java code :

```
//Java program to find greatest of three numbers
import java.util.Scanner;
public class greatest_of_three_numbers
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input three numbers from user
                System.out.print("Enter the first
number : ");
                int first = sc.nextInt();
                System.out.print("Enter the second
number : ");
                int second = sc.nextInt();
                System.out.print("Enter the third
number : ");
                int third = sc.nextInt();
                System.out.println();
                //condition for first number
                if(first > second && first > third)

                        System.out.println(first+" is
the greatest number.");
                //condition for second number
                else if(second > first && second >
third)
                        System.out.println(second+"
is the greatest number.");
                //condition for third number
                else if(third > first && third > second)

                        System.out.println(third+" is
the greatest number.");
                //condition when all three numbers are
equal
                else


                        System.out.println("All three
numbers are same");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();


        }
}
```

Python code :

```
first = int(input("Enter first number:"))
second = int(input("Enter second number:"))
third = int(input("Enter third number:"))

if first > second and first > third:
  print("First is Greater than Second and Third")
elif second > first and second > third:
  print("Second is Greater than First and Third")
else:
  print("Third is Greater than First and Second")

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 8. Leap year or not

In this program we have to find whether the year is a leap year or not. Generally we assume that year is exactly divisible by 4 is a leap year. But it is not only in this case 1900 is divisible by 4. But it is not a leap so it that case we follows these conditions

*It is exactly divisible by 100

*If it is divisible by 100, then it should also exactly divisible by 4

*And it is divisible by 400

These all conditions are true: a leap year is a leap year.

Working

Step 1. Initialize variable "year" to find leap year.

Step 2. Take input from User.

Step 3. We use this condition ((year%4==0)&&(year%100!=0)) || (year%400==0)) to check if the year is Leap or not.

Step 4. It is true display year is a leap year.

Step 5. The false display year is not a leap year.

Step 6. Stop.

C Code :

```
#include<stdio.h>
  int main()
  {
    //initialization of Year
    int year;

  //to take user input
  printf("Enter Year for find leap year or not : ");
  scanf("%d",&year);

  //we use this statement for check leap year
  if(((year%4==0)&&(year%100!=0)) || (year%400==0))
  printf("%d is a Leap Year",year);

  //not leap year
  else
  printf("%d is not a Leap Year",year);
  return 0;

  }
```

```
  Output

  Enter Year for find leap year or not : 2012
  2012 is a leap Year

  Enter Year for find leap year or not : 1900
  1900 is not a leap Year
```

C++Code :

```
//C++ Program
  //Leap year or not
  #include<iostream>
  using namespace std;
  //main program
  int main()
  {
    //initialising variables
    int year;
    cout<<"Enter year to check: ";
    //user input
    cin>>year;
    //checking for leap year
    if( ((year % 4 == 0) && (year % 100 != 0)) || (year %
400==0) )
    {
      //input is a leap year
      cout<<year<<" is a leap year";
    }
    else
    {
      //input is not a leap year
      cout<<year<< " is not a leap year";
    }
    return 0;
  }
```

Java Code :

```
//Java program to check whether a year entered by user is a
leap year or not
import java.util.Scanner;
public class LeapYear
{
        public static void main(String[] args)
        {
                //scanner class declaration
        Scanner sc=new Scanner(System.in);
                //input year from user
                System.out.println("Enter a Year");
                int year = sc.nextInt();
```

```
                    //condition for checking year entered
by user is a leap year or not
          if((year % 4 == 0 && year % 100 != 0) || year %
400 == 0)
               System.out.println(year + " is a leap year.");
                    else
                              System.out.println(year + " is
not a leap year.");
                    //closing scanner class(not compulsory,
but good practice)
                    sc.close();
      }
}
```

Python Code :

```
year = int(input("Enter Year:"))
if year % 4 == 0:
   if year % 100 == 0:
      if year % 400 == 0:
         print("Yes, {} is Leap Year".format(year))
      else:
         print("No, {} is Leap Year".format(year))
   else:
      print("No, {} is Leap Year".format(year))
else:
   print("No, {} is Leap Year".format(year))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 9. Prime number

A number is considered a prime number when it satisfies the below conditions.

Prime number is a number which can be divided by 1 and itself
A number which can not be divided by any other number other than 1 or itself is a prime number.
It should have only 2 factors. They are, 1 and the number itself.

Problem Description

In this program we will find if a number is a prime number or not with the help of a for loop or if else statement. A number is considered a prime number when it satisfies the below conditions.

- Prime number is a number which can be divided by 1 and itself
- A number which can not be divided by any other number other than 1 or itself is a prime number.
- It should have only 2 factors. They are, 1 and the number itself.

Ex- Number is 13. it have only 2 factor

- it is divisible by 1.
- And it is divisible by itself

So it is a prime number.

Working

Step 1. Read a “num” value to check prime or not.

Step 2. set i=1,div=0.

Step 3. if i<=num if true go to step 4, else go to step 7.

Step 4. Check the condition num%i==0 if true then

evaluate step 5, else go to step 6.

Step 5. set div=div+1.

Step 6. i=i+1, go to step 4.

Step 7. check div, if div==2 display prime, else display not

prime.

Step 8. Stop

C Code :

```
#include<stdio.h>
int main()
{
    //initializing variables
    int c,number,div=0;

//user input
printf(“Enter number: “);
scanf(“%d”,&number);

//checking for number of divisor
for(c=1;c<=number;c++)
{
if(number%c==0)
{
div++;
}
}
//no divisors other than 1 and itself
if(div==2)
{
//display
printf(“%d is a prime number”,number);
}
else
{
//display
printf(“%d is not a prime number”,number);
}
return 0;
}
```

Output

```
Enter Number:6
6 is not a Prime Number

Enter Number:13
13 is a Prime Number
```

C++ Code :

```
//C++ Program
//Check Prime or Not
#include<iostream>
using namespace std;
int main()
{
   int i,num,div=0;        //initializing variables
   cout<<“Enter number:”;
   cin>>num;          //user input
   for(i=1;i<=num;i++)     //checking for number of divisor
   {
      if(num%i==0)
      {
         div++;
      }
   }
   if(div==2)          //no divisors other than 1 and itself
   {
      cout<<num<<” is a prime number”;
   }
   else
   {
      cout<<num<<” is not a prime number”;
   }
   return 0;
}
```

Java Code :

```
//JAVA Program to check whether the number entered by
user is Prime or not.
import java.util.Scanner;
public class prime
{                                          //class declaration
      public static void main(String[] args)
            {                                         //main
method declaration
            Scanner sc=new Scanner(System.in);
//scanner class object creation

                        System.out.println("Enter a number");
                        int n = sc.nextInt();
//taking a number n as input
                        int count=0;
                        for(int i = 1 ; i <=n ; i++)
                        {
                                    if(n % i == 0)
                        //condition for getting the factors of
number n
                                    count=count+1;
                        }
                        if(count == 2)
//if factors are two then, number is prime else not
                        System.out.println("Prime Number");
                        else
                        System.out.println("Not a Prime
Number");
                        sc.close();
//closing scanner class(not mandatory but good practice)
            }                                   //end of main
method
}                                         //end of class
```

Output :

PythonCode :

```
a = 0
count = 0
n=int(input("Enter the number to check if it is prime or not:
"))
a = n // 2;
for i in range(2,a+1):
   if (n % i == 0):
      print("The given number is not prime")
      count = 1
      break
if (count == 0):
   print("The given number is prime")
```

Chapter 10. Prime number within a given range

A number that is divisible only by itself and 1 (e.g. 2, 3, 5, 7, 11).

The C program reduces the number of iterations within the for loop. It is made to identify or calculate the prime numbers within a given range of numbers inserted by the user.

Ex:- if a user enters a range as 40-50 In that range 41,43,47 these three numbers are prime numbers.

Problem Description

The C program reduces the number of iterations within the for loop. It is made to identify or calculate the prime numbers within a given range of numbers inserted by the user. The program takes the range and identifies all the prime numbers between the given range as well as similarly prints the digits coming under the prime numbers. Users are required to take the range as input that will be stored in the variables num1 and num2 respectively.

Ex:- if a user enters a range as 40-50 In that range 41,43,47 these three numbers are prime numbers.

Working

Step 1: Start

Step 2: The user is asked to insert a given range of numbers as an input to find the prime numbers.

Step 3: Find prime numbers within the given range that should be only odd values.

Step 4: Check whether the odd numbers are divisible by any of the natural numbers

Step 5: Print the calculated prime numbers.

Step 6: Stop

C Code :

```
#include<stdio.h>
   #include<stdlib.h>
   void main()
   {
   //To initialize variables
      int num1, num2, i, j, flag, temp, count = 0;
   //for taking user input
      printf("Insert the value of num1 and num2 \n");
      scanf("%d %d", &num1, &num2);
   //check condition first range is less than 2
      if (num2 < 2)
      {
         printf("No prime nums found up-to %d\n", num2);
         exit(0);
      }
   //to display prime numbers
      printf("Prime nums are \n");
      temp = num1;
   //if num1 modules 2 is equal to zero
   if( num1 % 2 == 0)
      {
   //increment on that number.
         num1++;
      }
   //use for loop with first rang and second rang
      for (i = num1; i <= num2; i = i + 2)
      {
         flag = 0;
         for (j = 2; j <= i / 2; j++)
         {
            if ((i % j) == 0)
            {
               flag = 1;
               break;
            }
         }
   //check if flag equal to zero
         if (flag == 0)
         {
         //display
            printf("%d\n", i);
            count++;
         }
      }
   //display total prime number b/w lie on given range
      printf("Num of primes between %d & %d = %d\n",
temp, num2, count);
   }
```

```
   Output

   Insert the value of num1 and num2:
   70, 80
   Prime nums are
   71
   73
   79
   83
   Num of primes between 70 and 85 = 4
```

C++ Code :

```
//C++ Program

   //Prime numbers in a given range

   #include<iostream>

   using namespace std;

   //function to chek for prime number

   void prime(int num)

   {

      int div=0;

      //checking for number of divisor

      for(int i=1;i<=num;i++)

      {

         if(num%i==0)

            div++;

      }

      //no divisors other than 1 and itself

      if(div==2)
```

```
        cout<<num<<endl;

    }

    int main()

    {

      cout<<"Enter range:";

      int lowerLimit, upperLimit;

      //user input

      cin>>lowerLimit>>upperLimit;

      cout<<"Prime numbers between "<<lowerLimit<<"
and "<<upperLimit<<" are:"<<endl;

      //finding prime numbers in the given range

      for(int i=lowerLimit;i<=upperLimit;i++)

        prime(i);

      return 0;

    }
```

Java Code :

```
//Java program to print prime numbers in a given range
import java.util.Scanner;
public class prime_numbers_in_a_given_range
{
        public static void main(String[] args)
        {
                //scanner class object creation
                Scanner sc=new Scanner(System.in);
                //input from user
                System.out.print("Enter Starting
Number : ");
                int start = sc.nextInt();
                System.out.print("Enter Ending
Number : ");
                int end = sc.nextInt();
                System.out.println("Prime numbers
between "+start+" and "+end+" are : ");
                int count;
                //loop for finding and printing all prime
numbers between given range
                for(int i = start ; i <= end ; i++)
                {
                        //logic for checking number
is prime or not
                        count = 0;
                        for(int j = 1 ; j <= i ; j++)
                        {
                                if(i % j == 0)
                                        count =
count+1;
                        }
                        if(count == 2)

System.out.println(i);
                }
                //closing scanner class(not mandatory
but good practice)
                sc.close();
        }
}
```

Python Code :

```
first = int(input("Enter the first number:"))
second = int(input("Enter the Second Number:"))
for i in range(first, second):
    for j in range(2, i//2):
        if i % j == 0:
            break
    else:
        print("Prime Number", i)

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 11. Sum of digits of a number

This program in C programming calculates the sum of numbers inserted by the user or in an inserted integer. The program is taken as an input and stored in the variable number, denoted as no. Initially, the sum of the variable is zero, and then it is divided by 10 to obtain the result or output.

In this C program to allow the user enter any number and then it will divide the number into individual digits and add those individuals (Sum=sum+digit) digits using While Loop.

Ex:- number is 231456

2+3+1+4+5+6=21

sum of digit of a given number is 21

Working:-

Step 1: Start

Step 2: Ask the user to insert an integer as an input.

Step 3: Divide the integer by 10 in order to obtain quotient and remainder.

Step 4: Increase the new variable with the remainder received in the above step

Step 5: Repeat the above steps with the quotient till the value of the quotient becomes zero.

Step 6: Printf the output or sum

Step 7: Stop

C code :

```
/* C program to take a number & calculate the sum of its numbers */
#include<stdio.h>
int main()


{
  int no, temp, digit, sum = 0;

    printf ("Insert a number \n");
    scanf ("%d", &no);

    temp = no;
    while  (no > 0)
    {
        digit = no % 10;
        sum  = sum + digit;
        no /= 10;
    }
    printf("Given number = %d\n", temp);
    printf("Sum of the numbers %d = %d\n", temp, sum);
  return 0;
}
```

```
Output
Insert a number: 16789

Given number: 16789

Sum of the numbers: 31
```

C++ code :

```
 //C++ Program
 //Sum of digits in a number


    #include
    using namespace std;
    int main()
    {
        int num,sum=0;
        cout<<"Enter any num : ";
        //user input
        cin>>num;
        //loop to find sum of digits
        do
        {
```

```
        sum+=num%10;
        num=num/10;
    }while(num!=0);
    //output
    cout<<"\nSum of digits in given integer is: "<<sum;
    return 0;
}
```

Java Code

```
//Java program to calculate sum of digits of a number
import java.util.Scanner;
public class sum_of_digits
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
                //declare a variable to store sum of
digits
                int sod = 0;
                while(number != 0)
                {
                        int pick_last = number % 10;
                        sod = sod + pick_last;
                        number = number / 10;
                }
                //display sum of digits
                System.out.print("Sum of Digits =
"+sod);
                //closing scanner class(not compulsory,
but good practice)
                 sc.close();
     }
}
```

Python Code :

```
num = [int(d) for d in input("Enter the Number:")]
sum = 0
for i in range(0, len(num)):
   sum = sum + num[i]

print("Sum of Digits of a Number: {}".format(sum))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 12. Reverse of a number

In this program reverses a number entered by a user and then prints it. For example, if a user will enter 6577756 as input then 6577756 will be printed as output.

This C program accepts an integer and reverses it.

Working:-

Step 1. Take the number which you have to reverse as the input variable says number.

Step 2. Obtain its quotient and remainder.

Step 3. Multiply the separate variable with 10 and add the obtained remainder to it.

Step 4. Do step 2 again for the quotient and step 3 for the remainder obtained in step 4.

Step 5, Repeat the process until the quotient becomes zero.

Step 6.When it becomes zero, print the output and exit

Step 7. Stop.

C code :

```
#include<stdio.h>
int main()
{
   //Initialization of variables where rev='reverse=0'
   int number, rev = 0,store, left;

   //input a numbers for user
     printf("Enter the number\n");
     scanf("%d", &number);

    store= number;

    //use this loop for check true condition
    while (number > 0)
    {
     //left is for remider are left
      left= number%10;

     //for reverse of no.
      rev = rev * 10 + left;

      //number /= 10;
      number=number/10;

    }
   //To show the user value
    printf("Given number = %d\n",store);

    //after reverse show numbers
     printf("Its reverse is = %d\n", rev);

   return 0;
}
```

Output:-

```
Enter the number 123456

Given number = 123456
Its reverse is =654321
```

C Code

```
 //C++ Program
   //Reverse of a number
   #include <iostream>
   using namespace std;
   //main program
   int main()
   {
      //variables initialization
      int num, reverse=0, rem;
      cout<<"Enter a number: ";
      //user input
      cin>>num;
      //loop to find reverse number
      do
      {
         rem=num%10;
         reverse=reverse*10+rem;
         num/=10;
      }while(num!=0);
      //output
      cout<<"Reversed Number: "<<reverse;
   return 0;
   }
```

Java Code

```
//Java program to print reverse of a number
import java.util.Scanner;
public class reverse_of_number
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
                System.out.print("Reverse of
"+number+" is ");
                int reverse = 0;
                String s = "";
                while(number != 0)
                {
                        int pick_last = number % 10;
                        //use function to convert
pick_last from integer to string
                        s = s +
Integer.toString(pick_last);
                        number = number / 10;
                }
                //display the reversed number
                System.out.print(s);
                //closing scanner class(not compulsory,
but good practice)
                sc.close();


        }
}
```

Python Code

```
num = int(input("Enter the Number:"))
temp = num
reverse = 0
while num > 0:
   remainder = num % 10
   reverse = (reverse * 10) + remainder
   num = num // 10

print("The Given number is {} and Reverse is
{}".format(temp, reverse))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 13. Palindrome number

A palindrome number is a number that is given the same number after reverse. In C programs to check if the input number is palindrome or not. We are using a while loop and an else if statement in the C Program.

Ex:- A number is 123321 .If you read the number “123321” from reverse order, it is the same as “123321”.

In that number is a palindrome.

A number is 12121. If we read the number “12121” from reverse order ,it is the same as 12121. It is also a palindrome number

Working:-

Step 1.Take the number which you have to reverse and find the palindrome as the input variable says number.

Step 2.Number is stored in his duplicity value as a duplicate variable (n1).

Step 3.Obtain its quotient and remainder.

Step 4.Multiply the separate variable with 10 and add the obtained remainder to it.

Step 5.Do step 2 again for the quotient and step 3 for the remainder obtained in step 4.

Step 6.Then we check if reverse is equal to a number.

Step 7.Its true display number is palindrome

Step 8.It is false display number is not a palindrome.

Step 9.stop.

C Code

```
#include<stdio.h>
int main()
{
   //Initialization of variables where rev='reverse=0'
    int number, rev = 0,store, n1,left;

    //input a numbers for user
    printf("Enter the number\n");
    scanf("%d", &number);

    //for duplicacy of number
     n1=number;

     store= number;
     //use this loop for check true condition
     while (number > 0)
     {
         //left is for remider are left
          left= number%10;

          //for reverse of no.
           rev = rev * 10 + left;

           //number /= 10;
            number=number/10;
       }
       //To check reverse no is a Palindrome
       if(n1==rev)
          printf("Number %d is Palindrome number",n1);
       else
          printf("it is not a Palindrome number");
   return 0;
}
```

```
Output:-
Enter the number

121121

Number 121121 is Palindrome number.
```

C++ Code

```
//C++ Program
//Palindrome or not
#include <iostream>
using namespace std;
//main Program
int main()
{
   int num, digit, reverse = 0;
   cout << “Enter a positive integer: “;
   //user input
   cin >> num;
   int temp = num;
   //loop to find reverse
   do
   {
      digit = num % 10;
      reverse = (reverse * 10) + digit;
      num = num / 10;
```

```
    } while (num != 0);
    cout << “The reverse of “<< temp <<” is “<< reverse
<< endl;
    //checking for palindrome
    if (temp == reverse)
        cout << “The number is a palindrome.”;
    else
        cout << “The number is not a palindrome.”;
    return 0;
  }
```

Java Code

```
//Java program to check whether a string entered by user is
palindrome or not.
import java.util.Scanner;
public class palindrome_or_not
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a String : ");

                String st = sc.next();
                //string function for calculating length
of the string
                int len = st.length();
                //string variable to store reversed string
                String st1 = "";
                for(int i = 0 ; i < len ; i++)
                {
                        //string function for getting
character at a particular index
                        char ch = st.charAt(i);
                        st1 = ch + st1;
                }
                //condition for checking palindrome by
using string function
                if(st.equals(st1))

System.out.print("Palindrome");
                else
                        System.out.print("Not
Palindrome");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();


        }
}
```

Python Code

```
number = int(input("Enter the Number:"))
temp = number
reverse = 0
while number > 0:
    remainder = number % 10
    reverse = (reverse * 10) + remainder
    number = number // 10

if temp == reverse:
    print("Given number {} is Palindrome".format(temp))
else:
    print("Given number {} is not
Palindrome".format(temp))

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student
```

Chapter 14. Armstrong number

In this program we will find the number is Armstrong or not where the number should be entered by the user. Basically the sum of the cube of its digits is equal to the number itself is called Armstrong number.

Ex:- Enter any number 153.

1**3 + 5**3 + 3**3 = 153

Number is Armstrong

Working:

Step 1.Initialize variables num,n,n1,c=0,mul=1,sum=0,r,f,i .

Step 2.Input any number by user so read num variable.

Step 3.set n=num and n1=num for duplicate.

Step 4.We use while loop with condition(n!=0).

Step 5.Than check the last digit of a number with condition is reminder(r)=number(n)%10.

Step 6.Than increment of other variables for next step (c++).

Step 7.Than find length of number with condition is number(n)=number(n)/10.

Step 8.repeat steps 4 to 6 until number (n)!=0.

Step 9.Again we use the while loop with condition (num!=0) for check

Step 10.the number is Armstrong or not.

Step 11.Again check last digit for duplicity of number with condition is reminder(r1)=number(num)%10.

Step 12.Than we use for loop statement with condition is (i=1;i<=c).

Step 13.Use this code mul=mul*r1, sum=sum+mul, num=num/10;

Step 14.For find number is armstrong.

Step 15.The check if (n1==sum) display number is armstrong

Step 16.Either false display number is not armstrong

Step 17.stop

C Code

```
#include<stdio.h>
int main()
{
     int num ,n,n1,c=0,mul=1,sum=0,r,f,i;
     printf("enter any num: \n");
     scanf("%d",&num);
     n=num;
     n1=num;
     while(n!=0)
     {
        r=n%10;
        c++;
        n=n/10;
     }
    while (num!=0)
    {
       f=num%10;
       mul=1;
       for(i=1;i<=c;i++)
       {
           mul=mul*f;
       }

      sum=sum+mul;
     num=num/10;
    }
    if(n1==sum)
       printf("Armstrong Number");
   else
       printf("Not an Armstrong Number");
 return 0;
}
Output:-
enter any num: 1634

Armstrong Number.


enter any num: 135
```

C ++ Code

```
//C++ Program
   //Armstrong number or not
```

```
#include<iostream>
#include<math.h>
using namespace std;
//main Program
int main()
{
  int num, digit, sum = 0;
  cout << “Enter a positive  integer: “;
  //user input
  cin >> num;
  int store = num;
  //find sum of cubes of individual digits
  do
  {
    digit = num % 10;
    sum = sum + pow(digit,3);
    num = num / 10;
  }while(num != 0);
  //checking for ArmStrong number
  if(sum == store)
    cout << store << ” is an Armstrong number.”;
  else
    cout << store << ” is not an Armstrong number.”;
  return 0;
}
```

Java Code

```
//Java program to check whether a number is armstrong or
not
import java.util.Scanner;
public class armstrong_number_or_not
{
		public static void main(String[] args)
		{
				//scanner class declaration
				Scanner sc = new Scanner(System.in);
				//input from user
				System.out.print("Enter a number : ");

				int number = sc.nextInt();
				int n = number;
				int sum = 0;
				while(n != 0)
				{
						int pick_last = n % 10;
						sum = sum + (pick_last *
pick_last * pick_last);
						n = n / 10;
				}
				//condition for checking that the sum is
equal to the number or not
				if(sum == number)

System.out.println("Armstrong Number");
				else
						System.out.println("Not an
Armstrong Number");
				//closing scanner class(not compulsory,
but good practice)
				sc.close();


		}
}
```

Python Code

```
import math
value = int(input("Enter the Number: "))
num = [int(d) for d in str(value)]
sum = 0
for i in range(0, len(num)):
  sum = sum + math.pow(num[i], len(num))

if sum == value:
  print("Given number is Armstrong Number")
else:
  print("Given Number is not Armstrong Number")

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 15. Armstrong number in a given range

To identify the Armstrong number between two intervals in C programming, the user is required to insert integer numbers. A n digit number is known as an Armstrong number, when the sum of the values of the digits raised to nth power is equal to the number itself. For example: 153 = 13+53+33=153

Ex:- basically we know that Armstrong numbers in the given range 0 to 999 are 1 2 3 4 5 6 7 8 9 153 370 371 407.

Working

Step 1: Start

Step 2: Insert the start and end value

Step 3: Repeat from I = start value to end value

Step 4: Repeat the process until the temporary number is not equal to 0.

Step 5: Remainder = temp%0

Step 6: The result should be equal to the result plus the power (remainder n).

Step 7: temp = temp/10

Step 8: If the value of the result is equals to the value of number then print the number

Step 9: Else repeat the steps until the end number is encountered.

C Code

```
#include<stdio.h>
int main()
{
    //For initializing variables
     int start, end, i, temp1, temp2, rem, n = 0, result = 0;
    //user give start and end point of a number
   printf("Insert the start value and end value :");
   scanf("%d %d", &start, &end);

   //to display pint of range
   printf("\n Armstrong nums between %d an %d are: ",
start, end);

    //for use this loop to store all number in given range
   for(i = start + 1; i < end; ++i)
   {
       //store a duplicity value of given range
       temp2 = i;
       temp1 = i;

     while (temp1 != 0)
     {
         //temp1 /= 10;
        temp1=temp1/10;
         ++n;
     }
     while (temp2 != 0)
     {
       rem = temp2 % 10;
       //result += pow(rem, n);
      result=result+pow(rem,n);
     //temp2 /= 10;
       temp2=temp2/10;
     }
     //check true condition if result is equal to i
       if (result == i)
       {
          //display
          printf("%d ", i);
       }
       n = 0;
     result = 0;
   }
   printf("\n");
  return 0;
}
```

Output:
Insert the start value and end value: 100, 500
Armstrong numbers between 100 and 500 are: 370, 371, 407.

C ++ Code

```
   //C++ Program
   //Armstrong number in a interval
   #include<iostream>
   #include<math.h>
   using namespace std;
   void armstrong(int num)
   {
      int sum=0;
      int store = num;
```

```
    //find sum of cubes of individual digits
    do
    {
      int digit = num % 10;
      sum = sum + pow(digit,3);
      num = num / 10;
    }while(num > 0);
    //checking for ArmStrong number
    if(sum == store)
      cout << store <<"\t";
  }
  int main()
  {
    int l_limit,u_limit;
    cout<<"Enter the range:\n";
    cin>>l_limit>>u_limit;
    cout<<"Armstrong numbers between "<<l_limit<<"
and "<<u_limit<<" are:\n";
    for(int i=l_limit;i<=u_limit;i++)
      armstrong(i);
    return 0;
  }
```

Java Code

```
//Java program to print armstrong numbers between two
intervals
import java.util.Scanner;
public class armstrong_numbers_between_two_intervals
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter Starting
Number : ");
                int start = sc.nextInt();
                System.out.print("Enter Ending
Number : ");
                int end = sc.nextInt();
                System.out.println("Armstrong
numbers between "+start+" and "+end+" are : ");
                int n, sum;
                //loop for finding and printing all prime
numbers between given range
                for(int i = start ; i <= end ; i++)
                {
                        n = i;
                        sum = 0;
                        //logic for checking number
is armstrong or not
                        while(n != 0)
                        {
                                int pick_last = n %
10;
                                sum = sum +
(pick_last * pick_last * pick_last);
                                n = n / 10;
                        }
                        if(sum == i)

System.out.println(i);
                }
                //closing scanner class(not compulsory,
but good practice)
                sc.close();


        }
}
```

Python code

```
import math

first = int(input("Enter first number:"))
second = int(input("Enter second number:"))


def is_Armstrong(val: int) -> bool:
  sum = 0
  arr = [int(d) for d in str(val)]
  for i in range(0, len(arr)):
    sum = sum + math.pow(arr[i], len(arr))
  if sum == val:
    print("{} number is Armstrong".format(val))
  else:
    print("{} number is not Armstrong".format(val))

for i in range(first, second + 1):
  is_Armstrong(i)

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 16. Fibonacci Series upto nth term

The sequence is a Fibonacci series where the next number is the sum of the previous two numbers. The first two terms of the Fibonacci sequence start from 0,1,…

Example: limit is Fibonacci series 8

Sequence is 0,1,1,2,3,5,8,13

It's followed by an additional operation. Next number is the addition of before the first two numbers.

Working

Step 1.Initialize variables limit, N1=0, N2=1, N3, i.

Step 2.To take user input for limit of seriousness.

Step 3.Display N1, N2 value .

Step 4.We use a loop with the condition(i=0;i<limit).

Step 5.Compute N3 = N1 + N2 and n1=n2. && n2=n3.

Step 6.Than display N3 as output and close the loop

Step 7.Stop

C Code

```
#include<stdio.h>
int main()
{
    //To initialize variables
    int n1=0,n2=1,n3,limit,i;

    //To take user input
    printf("enter a limit of series \n");
    scanf( "%d",&limit);

    printf("Fibonacci series %d %d ",n1,n2);

    //To use this loop for given length
    for(i=2;i<limit;i++)
    {
       //n1 and n2 sum  store in new variable n3
       n3=n1+n2;
       n1=n2;
       n2=n3;
       //display serious
        printf("%d ",n3);
     }
    return 0;
}
```

Output:-

```
enter a limit of series

10

Fibonacci series  0 1 1 2 3 5 8 13 21 34
```

C++ Code

```
//C++ Program
  //Fibonacci Series upto n numbers
  #include<iostream>
  using namespace std;
  //main program
  int main()
  {
  // initialising variables
  int limit, first=0, second=1, next, num;
  cout <<"Enter the limit of Fibonacci series"<<endl;
  // user input
  cin >> num;
  cout << "First "<<num<<" terms of Fibonacci series are
:- "<<endl;
  //loop for printing fibonacci series
  for(int p=0;p<num;p++)
  {
     if (p <= 1)
        next = p;
     else
     {
        next = first + second;
        first = second;
        second = next;
     }
     cout<<next<<" ";
  }
  return 0;
  }
```

Java Code

```
//Java program to print fibonacci series up to n
import java.util.Scanner;
public class fibonacci
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
        System.out.print("Enter the limit : ");

        int lim = sc.nextInt();
        if(lim > 0)
        {
            int y = 0, z = 1, s;
            //display starting two
numbers of series
            System.out.print("Fibonacci
Series : "+y+"  "+z+"  ");
            //perform iterations till the
limit entered by the user
            while(z <= lim)
            {
                s=y+z;
                y=z;
                z=s;
                //condition for
forcing z that it should not be printed when its value is
greater than limit
                if(z <= lim)

System.out.print(z+"  ");
            }
        }
        else
            System.out.print("Wrong
Input");
        //closing scanner class(not compulsory,
but good practice)
        sc.close();


    }
}
```

Python Code

```
# Method 1

def fibonacciSeries(i):
    if i <= 1:
        return i
    else:
        return (fibonacciSeries(i - 1) + fibonacciSeries(i - 2))


if num <= 0:
    print("Please enter a Positive Number")
else:
    print("Fibonacci Series:", end=" ")
    for i in range(num):
        print(fibonacciSeries(i), end=" ")

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
# Method 2

num = int(input("Enter the Number:"))
n1, n2 = 0, 1
print("Fibonacci Series:", n1, n2, end=",")
for i in range(2, num):
    n3 = n1 + n2
    n1 = n2
    n2 = n3
    print(n3, end=" ")

print()

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 17. Factorial of a number

In this program we will find the factorial of a number where the number should be entered by the user. Factorial is a sequence of a number whose multiply by all previous numbers.

Ex:- No is 5.

5x4x3x2x1=120

Factorial of a 5=120

Note:-Factorial of n number is 1*2*3*…n. You will learn to calculate the factorial of a number using for loop in this example.

Working

Step 1.Read the number n

Step 2.Initialize the variable i, fact=1,n

Step 3.To take a user input for factorial of number

Step 4.We use for loop with the condition(i=1;i<=number)

Step 5.Than do fact=fact*i

Step 6.Print the variable of fact.

Step 7.Stop

C Code

```
#include <stdio.h>
int main()
{
 //initialize of variable
   int i, number, fact = 1;

   //to take user input.
   printf("Enter a number to calculate its factorial\n");
   scanf("%d", &number);

   //use this loop of following statement
    for (i = 1; i<= number;i++)
      fact = fact * i;

    //display of factorial of a given number
   printf("Factorial of a number %d is = %d\n", number,
fact);

  return 0;
}
```

Output:-

```
Enter a number to calculate its factorial  4

Factorial of a number 4 is  24
```

C++ Code

```
 //C++ Program
   //Factorial of a number
   #include<iostream>
   using namespace std;
   //main program
   int main()
   {
      //initializing variables
      int fact=1,num;
      cout<<“Enter the number: “;
      //user input
      cin>>num;
      //checking for negative input
      if(num<0)
         cout<<“Invalid input!!\nEnter whole numbers
only”;
      // for positive numbers
      else
      {
         for(int i=num;i>0;i–)
         {
            fact*=i;
         }
         cout<<“Factorial of “<<num<<” is “<<fact;
      }
      return 0;
   }
```

Java Code

```
//Java program to find factors of a number
import java.util.Scanner;
public class factors_of_a_number
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
                System.out.println("Factors of
"+number+" are :");
```

```
                    //loop for finding factors of a number
                    for(int i = 1 ; i <= number ; i++)
                    {
                              if(number % i == 0)
                                        //printing factors

System.out.print(i+"              ");
                    }
                    //closing scanner class(not compulsory,
but good practice)
                    sc.close();

          }
}
```

Python Code

```
num = int(input("Enter the number:"))
factorial = 1
for i in range(1, num+1):
   factorial = factorial * i

print("Factorial of a Given Number:", factorial)
```

Chapter 18. Power of a number

In this program we will calculate the power of a number using C programming. We want to calculate the power of any given number so you need to multiply that particular number power of time.

Ex:-|

1. Let's suppose the number is 24 so we need to multiply with 4 times 2. That is 2*2*2*2=16.

2. Number is 53 so we need to multiply with 3 times 5. That is 5*5*5=125

Working:

Step 1– Enter the base number, the number in which you just want to find the power of the number.

Step 2– Enter the exponential, the power of the number.

Step 3– Initialize while loop, while the exponential is not equal to zero.

(i) do temp*number and store it in the temp.

(ii) now reduce the exponential with -1.

Here you got the power of the number.

Step 4– Now print the temp variable.|

Step 5- Stop.

C Code

```
#include<stdio.h>
int main()
{
   //To initialize variables
    int number, expo,temp = 1;

   //To take user input
    printf("Enter a base number: ");
    scanf("%d", &number);
   //To display Exponent
    printf("Enter an exponent: ");
    scanf("%d", &expo);
   //use while loop when power is not equal to zero
    while (expo != 0)
    {
      //temp*=number
       temp = temp * number;
       --expo;
    }
    printf("power of a %d is %d",number, temp);
   return 0;
}
```

Output:-

```
Enter a base number: 6
Enter an exponent: 4
power of a 6 is 1296
```

C++ Code

```
 //C++ Program
   //Power of a number
   #include <iomanip>
   #include <iostream>
   #include <math.h>
   using namespace std;
   //main program
   int main()
   {
      double exp, base;
      cout<<“Enter base: “;
      //user input 1
      cin>>base;
      cout<<“Enter Exponent: “;
      //user input 2
      cin>>exp;
      //calculating power using function
      double res = pow(base, exp);
      //printing result
      cout << base << “^” << exp << ” = “ ;
      cout << fixed <<setprecision(2)<<res<<endl;
      return 0;
   }
```

Java Code

```
//Java program to calculate power of a number
import java.util.Scanner;
public class Power_of_a_number
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input base value and exponent value
from user
                System.out.print("Enter the value of
base : ");
                int base = sc.nextInt();
                System.out.print("Enter the value of
exponent : ");
                int exp = sc.nextInt();
                //declare an integer variable to store the
result
        int result = 1;
                //logic for calculating power of the
entered number
        while (exp != 0)
        {
           result = result * base;
           --exp;
        }
                //print the result
        System.out.println("Answer = " + result);
                //closing scanner class(not compulsory,
but good practice)
                sc.close();
    }
}
```

Python Code

```
base = int(input("Enter Base number:"))
expo = int(input("Enter Expo Number:"))
temp = 1

for i in range(0, expo):
   temp = temp * base

print(temp)
```

Chapter 19 Factor of a number

In this Program we will calculate the factors of any numbers using C programming. The factors of a number are defined as the number we multiply two numbers and get the original number. The factor of a number is a real number which divides the original completely with zero remainder.

Ex- no is 16,5.

16= 2 x 2 x 2 x 2

5= 1 x 5

Working

Step 1- Enter the number, to find their factor.

Step 2- Initialise the loop with u=1 to u<=number and follow the following calculation

(i) check whether the number is divisible with u and u got a result zero.

(ii) now print the value of u.

From this loop u got all the factors of the number.

Step 3- Stop.

C Code

```
#include<stdio.h>
int main()
{
    //To initialize variable
    int number, u;
    //to take user input
    printf("Enter an any number: ");
    scanf("%d",&number);

    printf("Factors of a number %d are: ", number);

   //Use for loop this condition
    for(u=1; u<= number; u++)
    {
      //now we check for true condition of this
       if (number%u == 0)
         //display factor
          printf("%d ",u);
     }
   return 0;
}
```

```
Output:-
Enter an any number: 12
Factors of a number 12 are: 1 2 3 4 6 12
```

C ++ Code

```
//C++ Program
   //Factors of a number
   #include <iostream>
   using namespace std;
   //main Program
   int main()
   {
      int num;
      cout << “Enter a positive number: “;
      //user input
      cin >> num;
      cout << “Factors of “ << num << ” are: “ << endl;
      //finding and printing factors
      for(int i = 1; i <= num; i++)
      {
         if(num % i == 0)
            cout << i << “\t“;
      }
      return 0;
   }
```

Java Code

```
//Java program to find factors of a number
import java.util.Scanner;
public class factors_of_a_number
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input from user
                        System.out.print("Enter a number : ");

                        int number = sc.nextInt();
                        System.out.println("Factors of
"+number+" are :");
                        //loop for finding factors of a number
                        for(int i = 1 ; i <= number ; i++)
                        {
                                    if(number % i == 0)
                                                //printing factors

System.out.print(i+"              ");
                        }
                        //closing scanner class(not compulsory,
but good practice)
```

```
                sc.close();


        }
}
```

Python Code

```
number = int(input("Enter the Number:"))
for i in range(1, number+1):
    if number % i == 0:
        print(i, end=" ")

# This code is contributed by Shubhanshu Arya (Prepinsta
Placement Cell Student)
```

Chapter 20. Strong number

In this program we will find a strong number of not using C programming. Where the number should be entered by a user. We will use the While Loop ,for loop and else if statement in this program. In that program we use a user defined function for finding the factorial of number t + 5!=1 + 24 +120=145

what will be used to find strong numbers?

Basically A strong number is a number whose sum of factorials of digits is equal to the same number.Ex:- number is 145

1! + 4!

So it is a strong number.

Working:

Step 1- First we enter the number.

Step 2- copy the number into any temporary variable.

Step 3- until temp is not equal to 0, calculate the following statement

(i)digit=temp%10

(ii)now we find the factorial of a digit.

(iii)sum=sum+digit, add digit into a sum and store it in the sum.

(iv)temp=temp/10.divid the temp with 10

When temp became zero the above step stop to execute,

Step 4- if the sum of these factorial numbers is equal to the entered number,so it is the strong number.

Step 5- Stop.

C Code

```
#include<stdio.h>

//find factorial of a number.
int factorial(int number)
 {
     //to initialize of factorial
     int i,fact=1;
    //use for loop with this condition
    for(i=1;i<=number;i++)
    {
        //fact*=1;
        fact=fact*i;
    }
  return fact;
}
//to main function
int main()
{
      //to initialize variables
      int number,digit,sum=0,temp;

      //To take user input
      printf("Enter a number:");
      scanf("%d",&number);
      //To store a duplicity value of a given number
      temp=number;

     //use this whenever number is not equal to 0
    while(temp!=0)
    {
        //for last digit
        digit=temp%10;
       //now we call of factorial function
         digit = factorial(digit);
       //to improve of a sum on digit
         sum=sum+digit;
         temp=temp/10;
     }
     //we check sum is equal to number its true
     if(sum==number)
     {
        //display
        printf("It is a Strong Number");
     }
     //false condition
     else
     {
         //display
         printf("It is not Strong Number");
     }
   return 0;
}
```

Output:-

```
Enter a number: 145

It is a Strong Number


Enter a number: 123

It is not Strong Number
```

C ++ Code

```
 //C++ program
   //Strong Number or not
   #include<iostream>
   using namespace std;
   //main Program
   int main()
   {
      int ip,sum=0;
      cout<<"Enter number to check: ";
          //user input
      cin>>ip;
      int save=ip;
      //logic to check for Strong Number starts
      while(ip)
```

```
    {
      int num=ip%10;
      int fact = 1;
      //finding factorial of each digit of input
      for(int i=num;i>0;i–)
      {
        fact=fact*i;
      }
      sum+=fact;
      ip/=10;
    }
    //checking for Strong Nunber
    if(sum==save)
    {
      cout<<save<<" is a Strong Number";
    }
    else
    {
      cout<<save<<" is not a Strong Number";
    }
    //logic ends
    return 0;
  }
```

Java Code

```
//Java program to check whether a number is a strong
number or not
import java.util.Scanner;
public class strong_number_or_not
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
                int fac,sum = 0;
                int n = number;
                while(n != 0)
                {
                        fac = 1;
                        int r = n % 10;
                        //calculating factorial of r
                        for(int i = r ; i >= 1 ; i--)
                        fac = fac * i;
                        //storing sum of factorial of
all digits of the number
                        sum = sum + fac;
                        n=n/10;
                }
                //condition for strong number
                if(sum == number)
                                System.out.println("Strong
Number");//display the result
                else
                                System.out.println("Not a
Strong Number");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();


        }
}
```

Python Code

```
#Enter the number to check
print('Enter the number:')
n=int(input())
#save the number in another variable
temp=n
sum=0
#Implementation
while(temp):
  r=temp%10 # r will have the value of the unit place digit
  temp=temp//10
  fac=1
  for i in range(1,r+1): #finding factorial
    fac=fac*i

  sum+=fac #adding all the factorial

if(sum==n):
  print('Yes', n ,'is strong number')

else:
  print('No' , n, 'is not a strong number')
```

Chapter 21. Perfect number

In this program we will find the number is a perfect number or not using C programming. so we will use the while loop and if else statement. Basically perfect number is a positive number which is equal to the sum of all its divisors excluding itself. we have to find all divisors of that number and find their sum, if the sum of divisors is equal to number it means the number is Perfect Number. Else sum is not equal to number it means number is not a perfect number.

Ex:- Enter any number 6

6 is a perfect number as 1 + 2 + 3 = 6.

Number is 15

15 is not a perfect number because 1+3+5=9

Working:

Step 1- enter the number to be check

Step 2- initialize i with 1.

|Step 3- now execute the while loop, while i is less than the number so calculate the following expression.

(i) if number is divided by the i, so add number with the total and store it in total

(ii)increment the i with 1.

When i is equal to or greater than the number so loop will terminate.

Step 4- now compare the entered number with the total number.

Step 5- if the total number is equal to the entered number so the number is the perfect number.

Step 6- Stop

C Code

```
#include<stdio.h>
int main()
{
     // Initialization of variables
     int number,i=1,total=0;

    // To take user input
    printf("Enter a number: ");
    scanf("%d",&number);

   while(i<number)
    {
      if(number%i==0)
      {
       total=total+i;
       i++;
      }
    }
  //to condition is true
  if(total==number)
     //display
     printf("%d is a perfect number",number);
  //to condition is false
  else
     //display
      printf("%d is not a perfect number",number);

  return 0;
}
```

Output:-

```
Enter a number: 28

28 is a perfect number


Enter a number: 153

153 is not a perfect number
```

C++ Code

```
//C++ Program
  //Perfect Number or not
  #include<iostream>
  using namespace std;
  //main Program
  int main ()
  {
     int  div, num, sum=0;
     cout << “Enter the number to check : “;
     //user input
     cin >> num;
     //loop to find the sum of divisors
     for(int i=1; i < num; i++)
     {
        div = num % i;
        if(div == 0)
```

```
         sum += i;
      }
      //checking for perfect number
      if (sum == num)
         cout<< num <<" is a perfect number.";
      else
         cout<< num <<" is not a perfect number.";
      return 0;
   }
```

Java Code

```
//Java program to check whether a number is perfect or not
import java.util.Scanner;
public class perfect_number_or_not
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
                //declare a variable to store sum of
factors
                int sum = 0;
                for(int i = 1 ; i < number ; i++)
                {
                        if(number % i == 0)
                                sum = sum + i;
                }
                //comparing whether the sum is equal
to the given number or not
                if(sum == number)
                        System.out.println("Perfect
Number");
                else
                        System.out.println("Not an
Perfect Number");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();

                                }
}
```

Python Code

```
n = int(input("Enter any number: "))
sump= 0
for i in range(1, n):
   if(n % i == 0):
      sump= sump + i
if (sump == n):
   print("The number is a Perfect number")
else:
   print("The number is not a Perfect number")
```

Chapter 22. Automorphic number

In this program we have to find whether the number is an Automorphic number or not using C programming. An automorphic number is a number whose square ends with the same digits as number itself.

Automorphic Number in C Programming

Example:

- 5=(5)2=25
- 6=(6)2=36
- 25=(25)2=625
- 76=(76)2=5776
- 376=(376)2=141376

These numbers are automorphic numbers.

- Automorphic number : C | C++ | Java

Working

Step 1- Enter the number to be checked.

Step 2- store the number in a temporary variable.

Step 3- find the square of a given number and display it.

Step4- Initialize the while loop until the number is not equal to zero

(i) Calculate the remainder of the temp,divided with the 10 and store in digit

(ii) divide the number with the 10 and store in the number.

Step 5- find modules of square with count and compare with temp

Step 6-if it is true display Automorphic or else not a

Automorphic number

Step 7- Stop.

C Code

```
#include<stdio.h>

int checkAutomorphic(int num)
{
   int square = num * num;

   while (num > 0)
   {
      if (num % 10 != square % 10)
         return 0;

      // Reduce N and square
      num = num / 10;
      square = square / 10;
   }
   return 1;
}

int main()
{
   //enter value
   int num;
   scanf("%d",&num);

   //checking condition
   if(checkAutomorphic(num))
      printf("Automorphic");
   else
      printf("Not Automorphic");
   return 0;
}
```

Output:-

```
6
Automorphic

12
Not Automorphic

376
Not Automorphic
```

C++ Code

```
 //C++ Program
   //Automorphic number or not
   #include<iostream>
   using namespace std;
   //main program
   int main()
   {
      int num,flag=0;
      cout<<"Enter a positive number to check: ";
      //user input
      cin>>num;
      int sq= num*num;
      int store=num;
      //check for automorphic number
      while(num>0)
      {
         if(num%10!=sq%10)
         {
                  flag=1;
                  break;
               }
         num=num/10;
         sq=sq/10;
      }
      if(flag==1)
         cout<<store<<" is not an Automorphic number.";
      else
         cout<<store<<" is an Automorphic number.";
      return 0;
   }
```

Java Code

```
//Java program to check whether a number is Automorphic
number or not
import java.util.Scanner;
public class automorphic_number_or_not
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
        //Convert the number to string
                String s1 = Integer.toString(number);
        //Calculate the length
                int l1 = s1.length();
                int sq = number * number;
                String s2 = Integer.toString(sq);
                int l2 = s2.length();
        //Create Substring
                String s3 = s2.substring(l2-l1);
                if(s1.equals(s3))

System.out.println("Automorphic Number");
                else
```

```
                                        System.out.println("Not an
Automorphic Number");
                          //closing scanner class(not compulsory,
but good practice)
                          sc.close();


             }
}
```

Python Code

1st Approach

```
#enter the number to check
print('Enter the number:')
n=int(input())
sq=n*n #square of the given number
co=0 #condition variable
while(n>0): #loop until n becomes 0
   if(n%10!=sq%10):
      print('Not Automorphic')
      co=1
      break                # come out of the loop if the above
condition holds true
   #reduce n and square
   n=n//10
   sq=sq//10

if(co==0):
   print('Automorphic')
```

2nd Approach

```
n=int(input("Enter any number"))
x=n**2
a=str(n)
b=str(x)
y=len(a)
z=len(b)
if(z-b.find(a)==y):
   print("Automorphic")
else:
   print("Not automorphic number")
```

Chapter 23. Harshad number

In this program we will discuss if the number is harshad number or not in C programming. In mathematics, a Harshad number is a number that is divisible by the sum of its digits. We use a while loop statement with the following condition. Input consists of 1 integer.

Ex– Number is 21

it is divisible by own sum (1+2) of its digit(2,1)

So it is harshad number

Some other harshad numbers are 156,54,120 etc.

Working:

Step 1- Enter the number to be checked.

Step 2- store the number in a temporary variable.

Step 3- Initialise the while loop until the temp is not equal to zero

(i) Calculate the remainder of the temp,divided with the 10 and store in digit

(ii) add the digit with sum and store it in the sum.

(iii) divide the temp with the 10 and store in the temp;

Step 4- find modulus of the number with sum and store in the res;

Step 5- if res equal to zero then the given number is a harshad number else the given number is not a harshad number.

Step 6- Stop.

C Code

```
#include<stdio.h>
int main()
{
    //To initialize of variable
    int number,temp,sum = 0, digit, res;

   //To take user input
   printf("enter any number : ");
   scanf("%d",&number);

   //store in temporary variable
   temp = number;
  //use while loop with this condition
  while(temp!=0)
  {
     //to find last digit
      digit=temp % 10;
    //sum+=digit
      sum = sum + digit;
    //temp/=10
      temp = temp / 10;
  }
  res = number % sum;
  //check result is equal is to 0
  if(res == 0)
    //display
     printf("%d is Harshad Number",number);
  else
    //display
    printf("%d is not Harshad Number",number);
 return 0;
}
```

```
Output:-
enter any number : 21
21 is Harshad Number

enter any number : 15
15 is not  Harshad Number
```

C++ Code

```
 //C++ Program
   //Harshad number or not
   #include <iostream>
   using namespace std;
   //main program
   int main()
   {
      int num,sum = 0;
      cout<<"Enter number: ";
      //user input
      cin>>num;
      int n = num;
      //loop to calculate the sum of digits
      while(num > 0)
      {
         int rem = num%10;
         sum = sum + rem;
         num = num/10;
      }
      //checking for harshad number
      if(n % sum == 0)
         cout<<n<<" is a harshad number";
      else
         cout<<n<<" is not a harshad number";
      return 0;
      }
```

Java Code

```
//Java program to check whether a number is harshad
number or not
import java.util.Scanner;
public class harshad_number_or_not
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a number : ");

                int number = sc.nextInt();
                //make a copy of original number
                int n = number;
                //declare a variable to store sum of
digits
                int result = 0;
                //perform logic for calculating sum of
digits of a number
                while(n != 0)
                {
                        int pick_last = n % 10;
                        result = result + pick_last;
                        n = n / 10;
                }
                /*use condition to check whether the
number entered by
                user is completely divisible by its sum
of digits or not*/

                if(number % result == 0)
                        System.out.println("Harshad
Number");
                else
                        System.out.println("Not a
Harshad Number");
```

```
                        //closing scanner class(not compulsory,
but good practice)
                        sc.close();


            }
}
```

Output :

```
Enter a number : 18
Harshad Number


Enter a number : 345
Not a Harshad Number
```

List of Top 100 Cod

Python Code

General Solution:

```
n=int(input("Enter any number"))

p=n

l=[]

sum1=0

while(n>0):

   x=n%10

   l.append(x)

   n=n//10

sum1=sum(l)

if(p%sum1==0):

   print("Harshad number")

else:

   print("Not harshad number")
```

Optimal solution:

```
n=int(input("Enter any number"))

p=n

sum1=0

while(n>0):

   sum1+=n%10

   n=n//10
if(p%sum1==0):
   print("Harshad number")

else:
   print("Not harshad number")
```

Chapter 24. Abundant number

In this program to find the number is Abundant number or not. A number n is said to be Abundant Number to follow these condition

- the sum of its proper divisors is greater than the number itself.
- And the difference between these two values is called the abundance.

Ex:- Abundant number 12 having a proper divisor is 1,2,3,4,6 the sum of these factors is 16 it is greater than 12 so it is an Abundant number.

Some other abundant numbers

18, 20, 24, 30, 36, 66, 70, 72, 78, 80, 84, 88, 90, 96, 100, 102, 104, 108, 112, 114, 120..

Working

Step 1- Enter the number, to find the Abundant number.

Step 2- Initialize the loop with c=1 to c<=number and follow the following calculation

(i) check if the number is divisible with c and c got a result zero.

(ii) now sum=sum+c, add digit into a sum and store it in the sum.

Step 3. than check sum is greater than number print true.

Step 4. else it is not a abundant number

Step 5- Stop.

C Code

```
#include<stdio.h>

int main()
{

    //initialization variables

    int number,sum=0,c;

   //input from user

   printf("Enter a number : ");

    scanf("%d",&number);

  //declare a variable to store sum of factors of the number

   for(c = 1 ; c < number ; c++)

   {

        if(number % c == 0)
        //sum+=c;
         sum = sum + c;
   }
   if(sum > number)
       //display the result
        printf("Abundant Number");
   else
     //display
       printf("Not an Abundant Number");

  return 0;
}
```

C++ Code

```
//C++ Program
   //Abundant Number or not
```

```
#include<iostream>
using namespace std;
//main Program
int main ()
{
   int  div, num, sum=0;
   cout << “Enter the number to check : “;
   //user input
   cin >> num;
   //loop to find the sum of divisors
   for (int i=1; i < num; i++)
   {
   div = num % i;
      if (div == 0)
      sum += i;
   }
   //checking for Abundant number
   if (sum > num)
      cout<< num <<” is an Abundant number.”;
   else
      cout<< num <<” is not an Abundant number.”;
   return 0;
}
```

Java code

```
//Java program to check whether a number is abundant
number or not
import java.util.Scanner;
public class abundant_number_or_not
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input from user
                        System.out.print("Enter a number : ");

                        int number = sc.nextInt();
                        //declare a variable to store sum of
factors of the number
                        int sum = 0;
                        //loop for calculating sum of factors of
the number
                        for(int i = 1 ; i < number ; i++)
                        {
                                    if(number % i == 0)
                                                sum = sum + i;
                        }
                        //condition for checking whether the
sum is greater than number or not
                        if(sum > number)
                              System.out.println("Abundant
Number");
                        else
                              System.out.println("Not an
Abundant Number");
                        //closing scanner class(not compulsory,
but good practice)
                        sc.close();


            }
}
```

Output :

```
Enter a number : 12
Abundant Number
```

Python Code

```
#enter the number to check
print(‘Enter the number:’)
n=int(input())
sum=1 # 1 can divide any number
for i in range(2,n):
   if(n%i==0):    #if number is divisible by i add the
number
      sum=sum+i

if(sum>n):
   print(n,‘is Abundant Number’)

else:
   print(n,‘is not Abundant Number’)
```

Chapter 25. Friendly pair

Two numbers are said to be friendly pairs if they have a common abundancy index. Or, the ratio between the sum of divisors of a number and the number itself. These numbers are also known as Amicable numbers.

We can also say that two numbers n and m are friendly numbers if

?(n)/n = ?(m)/m

Where ?(n) is the sum of divisors of n.

For instance, for numbers 6 and 28,

Divisors of 6 are- 1, 2, 3, and 6.

Divisors of 28 are- 1, 2, 4, 7, 14, and 28.

Sum of the divisors of 6 and 28 are 12 and 56 respectively.

Also, the abundant index of 6 and 28 is 2.

Therefore, 6 and 28 is a friendly pair.

Working

Step 1. Start

Step 2. Input the numbers 1 and 2.

Step 3. Initialize two variables, sum1 and sum 2 with zero.

Step 4. Assign sum 1 with the sum of all the divisors of number 1.

Step 5. Assign sum 2 with the sum of all the divisors of number 2.

Step 6. If (sum 1==number1) and (sum 2==number 2), then print, “Friendly Numbers”

Step 7. Else print “Not Friendly Numbers”.

Step 8. Stop

C Code

```
#include<stdio.h>
int main()
{
   //1 Create two variables to use in first and second
numbers
     int i;
     int f_Num,s_Num;
   //2 two more variables created to store the sum of the
divisors
     int f_DivisorSum = 0;
     int s_DivisorSum = 0;

   //3 Asking user to enter the two numbers
      printf("Enter two numbers to check if Amicable or not
: ");
      scanf("%d %d",&f_Num,&s_Num);

   //4 Using one variable for loop and second to check for
each number
     for(int i=1;i<f_Num;i++)
     {
        //5 Condition check
        if(f_Num % i == 0)
           f_DivisorSum = f_DivisorSum + i;
     }
     //6 Calculating the sum of all divisors
    for(int i=1;i<s_Num;i++)
    {
       if(s_Num % i == 0)
          s_DivisorSum = s_DivisorSum + i;
    }
    //7 Check condition for friendly numbers
    if((f_Num == s_DivisorSum) && (s_Num ==
f_DivisorSum))

    else
    {
       printf("%d and %d are not Amicable
numbers\n",f_Num,s_Num);
    }
   return 0;
}
```

Output

```
Enter two numbers to check if Amicable or not : 12 13
12 and 13 are not Amicable numbers
```

C ++ Code

```
 //C++ Program
   //Friendly Pair(Amicable number) or not
   #include<iostream>
   using namespace std;
   // function to check Friendly pairs
   void findAmicable(int first, int second)
   {
      int sum1=0,sum2=0;
      for(int i=1; i<=first/2 ; i++)
      {
        //finding and adding divisors of first number
        if(first%i==0)
           sum1=sum1+i;
     }
     for(int i=1; i<=second/2 ; i++)
     {
        //finding and adding divisors of second number
        if(second%i==0)
           sum2=sum2+i;
     }
     //checking for friendly pair
     if(first==sum2 && second==sum1)
        cout<<"Friendly Pair("<<first<<","<<second<<")";
     else
        cout<<"Not a Friendly Pair";
  }
  //main program
  int main()
  {
     int first,second;
     cout<<"Enter first number : ";
     //user input
     cin>>first;
     cout<<"Enter second number : ";
     //user input
     cin>>second;
     //calling function
     findAmicable(first,second);
     return 0;
  }
```

Java Code

```
//Java program to check whether a number is friendly pair
or not
import java.util.Scanner;
public class friendly_pair_or_not
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter First number :
");
                int number1 = sc.nextInt();
                System.out.print("Enter Second
number : ");
                int number2 = sc.nextInt();
                //declare two variables to store the
addition of factors of both numbers which are entered by
user
                int add1 = 0, add2 = 0;
                //logic for finding factors and
calculating sum of all those factors for number1
                for(int i = 1 ; i < number1 ; i++)
                {
                        if(number1 % i == 0)
                                add1 = add1 + i;
                }
                //logic for finding factors and
calculating sum of all those factors for number2
                for(int i = 1 ; i < number2 ; i++)
                {
                        if(number2 % i == 0)
                                add2 = add2 + i;
                }
                //condition for friendly pair number
                if(number1 == add2 && number2 ==
add1)
                        System.out.println("Number
is Friendly Pair");
                else
                        System.out.println("Number
is not Friendly Pair");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();


        }
}
```

Python Code

```
#'Enter the numbers to check'
n=int(input())
m=int(input())
import math
sum_n=1 + n  #sum of divisor of n
sum_m=1 + m  #sum of divisor of m
i=2
j=2
#finding divisor
while(i<=math.sqrt(n)):
   if(n%i==0):
      if(n//i==i):
         sum_n+=i

      else:
         sum_n+=i + n//i

   i=i+1

while(j<=math.sqrt(m)):
   if(m%j==0):
      if(m//j==j):
         sum_m+=j

      else:
        sum_m+=j + m//j

    j=j+1
if(sum_n/n==sum_m/m):
    print(‘Yes’ , n , ‘,’ , m ,‘ are friendly Pair’)
else:
    print(‘No’, n , ‘,’ , m ,‘ are not friendly Pair’)
```

Chapter 26. Highest Common Factor(HCF):

The HCF or the Highest Common Factor of two numbers is the largest common factor of two or more values. The HCF can be calculated using some simple mathematical tricks. The following algorithm will determine how a c program can calculate the HCF of two numbers.

Working :-
Step 1. Start
Step 2. Define variables P and Q
Step 3. Develop a loop from 1 to the maximum value of P and Q.

Step 4. Check if both P and Q are completely divided by the same loop, if it does, store the number.
Step 5. Print the stored number as HCF.
Step 6. Stop

C Code :

```
#include <stdio.h>
int main()
 {
  //for initialize variables
   int a, b, i, hcf;
   a = 12;
   b = 16;
//find hcf of number
for(i = 1; i <= a || i <= b; i++)
 {
   if( a%i == 0 && b%i == 0 )
      hcf = i;
   }
 //display hcf
   printf("HCF = %d", hcf);
   return 0;
}
```

C++ Code :

```
 //C++ Program
   //GCD of Two Numbers
   #include<iostream>
   using namespace std;
   // Recursive function declaration
   int findGCD(int, int);
   // main program
   int main()
   {
          int first, second;
          cout<<"Enter First Number: ";
          cin>>first;
          cout<<"Enter second Number: ";
          cin>>second;
          cout<<"GCD of "<<first<<" and "<<second<<" is
"<<findGCD(first,second);
          return 0;
   }
   //body of the function
   int findGCD(int first, int second)
   {
          // 0 is divisible by every number
          if(first == 0)
          {
          return second;
          }
          if(second == 0)
          {
          return first;
          }
          // both numbers are equal
          if(first == second)
          {
          return first;
          }
          // first is greater
          else if(first > second)
          {
          return findGCD(first – second, second);
          }
          return findGCD(first, second – first);
   }
```

Java Code :

```
//Java program to find GCD or HCF of two numbers
import java.util.Scanner;
public class gcd_or_hcf
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input from the user
                        System.out.print("Enter the first
number : ");
                        int num1 = sc.nextInt();
                        //input from the user
System.out.print("Enter the second number : ");
                        int num2 = sc.nextInt();
                        int n = 1;
System.out.print("HCF of "+num1+" and "+num2+" is ");
                        if( num1 != num2)
                        {
                                    while(n != 0)
                                    {
                                    //storing remainder
                                    n = num1 % num2;

                                    if(n != 0)
                                    {
                                                num1 = num2;
                                                num2 = n;
                                                }
                                    }
                                    //result
                                    System.out.println(num2);

                        }
```

```
                    else
                    System.out.println("Wrong Input");
                    //closing scanner class(not compulsory,
but good practice)
                    sc.close();

          }
}
```

Python Code :

```
num1 = int(input("Enter first number:"))
num2 = int(input("Enter Second Number:"))
arr = []
if num1 > num2:
    smaller = num2
else:
    smaller = num1
for i in range(1,smaller+1):
    if (num1 % i == 0) and (num2 % i == 0):
        arr.append(i)
print("The HCF of given numbers: {}".format(max(arr)))
```

Chapter 27. Lowest Common Multiple (LCM) :

The Least Common Multiple (LCM) is also referred to as the Lowest Common Multiple (LCM) and Least Common Denominator (LCD). The least common multiple, or LCM, is another number that's useful in solving many math problems. Let's find the LCM of 12 and 44. One way to find the least common multiple of two numbers is to first list the prime factors of each number.

$12 = 2 \times 2 \times 3$

$44 = 2 \times 2 \times 11$

A C program can calculate the Lowest Common Multiple (LCM) of two numbers. The method includes finding out the maximum values among two numbers, which are common in both the numbers. The algorithm below will help to calculate the LCM of two numbers.

Working:-

- Initialize variable check1 and check2.
- Copy the value of n1 and n2 of variable .
- Initialize the while loop where condition is while(check1!=check2).

- In while loop there are two condition If check1<check2
- it is true use this condition check1=check1+n1; .
- Otherwise
- check2=check2+n2; .
- Print the value of check1 or check2.

C Code :

```
#include<stdio.h>

void lcm_two_no(int,int);
int main()
{
   int n1,n2;

  //to take user input n1,n2
   printf("Enter two numbers: ");
   scanf("%d %d",&n1,&n2);

   //call of user define function
   lcm_two_no(n1,n2);
   return 0;
}

//function to calculate l.c.m
void lcm_two_no(int n1,int n2)
{
  int check1,check2;
   //to use of duplicity value
   check1=n1;
   check2=n2;

    //to find lcm of number
    while(check1!=check2)
    {
       //for condition true
       if(check1< check2
           check1=check1+n1;

       //for condition false
        else
         check2=check2+n2;
     }
   printf("\nL.C.M of %d and %d is: %d",n1,n2,check1);

}
```

C++ Code :

```
//C++ program
   //LCM of two numbers
   #include<iostream>
   using namespace std;
   int findLCM(int,int);
   //main program
   int main()
   {
          int first,second;
          cout<<"Enter first number : ";
          cin>>first;
          cout<<"Enter second number : ";
          cin>>second;
          //calling function to find lcm
          cout<<findLCM(first,second)<<" is the LCM of
two numbers.";
          return 0;
   }
   //function to find lcm
   int findLCM(int first, int second)
   {
      static int fact = first;
      // if true then fact is the lcm
      if(fact % first == 0 && fact % second == 0)
      {
         return fact;
      }
      //if false call function again
      else
      {
         fact=fact + first;
         findLCM(first,second);
      }
      return fact;
   }
```

Java Code :

```
//Java program to find LCM of two numbers
import java.util.Scanner;
public class lcm
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input from the user
                        System.out.print("Enter the first
number : ");
                        int num1 = sc.nextInt();
                        //input from the user
                        System.out.print("Enter the second
number : ");
                        int num2 = sc.nextInt();
                        //logic for finding lcm of both numbers
```

```
        int i;
                int a =(num1 > num2)? num1 : num2;
                for(i = a ; i <= num1*num2 ; i=i+a)
                {
                        if(i % num1 == 0 && i %
num2 == 0)
                                break;
                }
                //printing result
                System.out.println("LCM of "+num1+"
and "+num2+" is : "+i);
                //closing scanner class(not compulsory,
but good practice)
                sc.close();

        }
}
```

Python Code :

```
num1 = int(input("Enter first number:"))
num2 = int(input("Enter Second Number:"))


def lcmFinder(num1, num2):
    if num1 > num2:
        larger = num1
    else:
        larger = num2
    while True:
        if (larger % num1 == 0) and (larger % num2 == 0):
            lcm = larger
            break
        larger = larger + 1
    print("LCM of two given number:{}".format(lcm))


lcmFinder(num1, num2)
```

Chapter 28. Greatest Common Divisor :

The Highest Common Multiple or the Greatest Common Divisor is the greatest number that exactly divides both numbers. It is possible to calculate this number through simple mathematical calculations. The following algorithm shows how the GCD of two numbers is calculated.

Ex:-

the H.C.F or G.C.D of 12 and 14 is 2.

The H.C.F or G.C.D of 16 and 12 is 4

Working:-
Step 1. Start

```
Step 2. Enter two numbers a and b.
Step 3. If a = 0, return b.
Step 4. If b = 0, return a.
Step 5. If a is equal to b return a
Step 6. If a is greater than b, return a – b, and b
Step 7. Else return a, b-a
Step 8. Stop
```

C Code :

```
// C program to calculate GCD of two numbers
#include<stdio.h>
// The code used a recursive function to return gcd of p and
q
 int gcd(int p, int q)
 {

   // checking divisibility by 0
    if (p == 0)
       return q;

    if (q == 0)
       return p;

    // base case
    if (p == q)
      return p;

   // p is greater
    if (p > q)
        return gcd(p-q, q);

    else
        return gcd(p, q-p);
 }

// Driver program to test above function
int main()
{
    int p = 98, q = 56;
    printf("GCD of %d and %d is %d ", p, q, gcd(p, q));
    return 0;
}
```

C++ Code :

```
//C++ Program
    //GCD of Two Numbers
    #include<iostream>
    using namespace std;
    // Recursive function declaration
    int findGCD(int, int);
    // main program
    int main()
    {
        int first, second;
        cout<<“Enter First Number: “;
        cin>>first;
        cout<<“Enter second Number: “;
        cin>>second;
        cout<<“GCD of “<<first<<” and “<<second<<” is
“<<findGCD(first,second);
        return 0;
    }
    //body of the function
    int findGCD(int first, int second)
    {
        // 0 is divisible by every number
        if(first == 0)
        {
        return second;
        }
        if(second == 0)
        {
        return first;
        }
        // both numbers are equal
        if(first == second)
        {
        return first;
        }
        // first is greater
        else if(first > second)
        {
        return findGCD(first – second, second);
        }
        return findGCD(first, second – first);
    }
```

Java Code :

```
//Java program to find GCD or HCF of two numbers
import java.util.Scanner;
public class gcd_or_hcf
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from the user
                System.out.print("Enter the first
number : ");
                int num1 = sc.nextInt();
                //input from the user
                System.out.print("Enter the second
number : ");
                int num2 = sc.nextInt();
                int n = 1;
                System.out.print("HCF of "+num1+"
and "+num2+" is ");
                if( num1 != num2)
                {
                        while(n != 0)
                        {
                                //storing remainder
                                n = num1 % num2;

                                if(n != 0)
                                {
                                        num1 =
num2;
                                        num2 =
n;
                                }
                        }
                        //result
                        System.out.println(num2);

                }
                else
                        System.out.println("Wrong
Input");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();

        }
}
```

Python Code :

```
num1 = int(input("Enter First Number:"))
num2 = int(input("Enter Second Number:"))


def gcdFunction(num1, num2):
  if num1 > num2:
    small = num2
  else:
    small = num1
  for i in range(1, small+1):
    if (num1 % i == 0) and (num2 % i == 0):
      gcd = i
  print("GCD of two Number: {}".format(gcd))

gcdFunction(num1, num2)
```

Chapter 29. Binary to Decimal to conversion :

The C program converts binary numbers to decimal numbers that are equivalent. A decimal number can be obtained by multiplying every digit of binary digit with power of 2 and totaling each multiplication outcome. The power of the integer starts from 0 and counts to n-1 where n is assumed as the overall number of integers in binary number.

Ex:- $(101100001)_2 = (353)_{10}$

To show on fig(1)

Working:-

Step 1: Start

Step 3: The user is asked to enter a binary number as an input

Step 4: Store the quotient and remainder of the binary number in the variable rem

Step 5: Multiply every digit of the entered binary number beginning from the last with the powers of 2 correspondingly

Step 6: Repeat the above steps with the quotient obtained until the quotient becomes 0

Step 7: The sum of the numbers will give the decimal number as a result, print the decimal val.

Step 8: Stop

C Code :

```
/** C program to convert the given binary number into
decimal**/
#include<stdio.h>
int main()
{
    int  num, binary_val, decimal_val = 0, base = 1, rem;

    printf("Insert a binary num (1s and 0s) \n");
    scanf("%d", &num); /* maximum five digits */

    binary_val = num;
    while (num > 0)
    {
      rem = num % 10;
      decimal_val = decimal_val + rem * base;
```

```
        //num/=10;
         num = num / 10 ;
        //base*=2;
         base = base * 2;
      }
     //display binary number
      printf("The Binary num is = %d \n", binary_val);
    //display decimal number
      printf("Its decimal equivalent is = %d \n",
decimal_val);
   return 0;
}
```

C++ Code :

```
//C++ Program
    //Convert binary to decimal
    #include <iostream>
    #include <math.h>
    using namespace std;
    //function to convert binary to decimal
    int convert(long n)
    {
        int i = 0,decimal= 0;
        //converting binary to decimal
        while (n!=0)
        {
            int rem = n%10;
            n /= 10;
            int res = rem * pow(2,i);
            decimal += res;
            i++;
        }
        return decimal;
    }
    //main program
    int main()
    {
        long binary;
        cout << "Enter binary number: ";
        cin >> binary;
        cout << binary << " in binary = " << convert(binary)
<< " in decimal";
        return 0;
    }
```

Java Code :

```
//Java program to convert Binary number to decimal
number
import java.util.Scanner;
public class Binary_To_Decimal
{
        public static void main(String args[])
        {
                Scanner sc = new Scanner(System.in);
                System.out.print("Enter a binary
number : ");
                int binary = sc.nextInt();
                //Declaring variable to store decimal
number
                int decimal = 0;
                //Declaring variable to use in power

                int n = 0;
                //writing logic for the conversion
                while(binary > 0)
                {
                        int temp = binary%10;
                        decimal +=
temp*Math.pow(2, n);
                        binary = binary/10;
                        n++;
                }
                System.out.println("Decimal number :
"+decimal);
        //closing scanner class(not compulsory, but good
practice)
                sc.close();
        }
}
```

Python Code :

```
num = int(input("Enter number:"))
binary_val = num
decimal_val = 0
base = 1
while num > 0:
    rem = num % 10
    decimal_val = decimal_val + rem * base
    num = num // 10
    base = base * 2

print("Binary Number is {} and Decimal Number is
{}".format(binary_val, decimal_val))
```

Chapter 30. Binary to Octal conversion :

Binary to octal conversion can be easily done with the help of simple calculations. The following section includes a stepwise procedure for such a conversion. In this process, a binary number is inputted by a user and is later converted to an octal number.

Working:
Step 1. Start
Step 2. Input a binary number
Step 3. Divide the number into groups of three bits.
Step 4. Multiply each bit from this group with the power of 2 and add them consecutively.
Step 5. Combine the results from all groups to generate the output.
Step 6. Print the octal number.
Step 7. Stop

C Code :

```
/** C Program to Convert Binary to Octal*/

#include<stdio.h>

int main()

{
 //For initialize variables
   long int binary_num, octal_num = 0, j = 1, rem;

 //Inserting the binary number from the user

   printf("Enter a binary number: ");
   scanf("%ld", &binary_num);

// while loop for number conversion

   while(binary_num != 0)
   {

      rem = binary_num % 10;
      octal_num = octal_num + rem * j;
     //j*=2
      j = j * 2;
     //binary_num/10;
      binary_num = binary_num / 10;

   }
   printf("Equivalent octal value: %ld", octal_num);

return 0;
}
```

C++ Code :

```
//C++ Program
   //binary to octal conversion
   #include <iostream>
   #include <math.h>
   using namespace std;
   //Function to convert binary to octal
   int convert(long binary)
   {
      int octal = 0, decimal = 0, i = 0,rem;
      //converting binary to decimal
      while(binary != 0)
      {
         rem = binary % 10;
         int res = rem * pow(2,i);
         decimal += res;
         i++;
         binary/=10;
      }
      i = 1;
      //converting decimal to octal
      while (decimal != 0)
      {
         rem = decimal % 8;
         octal += rem * i;
         decimal /= 8;
         i *= 10;
      }
      return octal;
   }
   //main program
   int main()
   {
      long binary;
      cout << “Enter a binary number: “;
      //user input
      cin >> binary;
      //calling function
      int octal=convert(binary);
      //printing output
      cout << binary << ” in binary = “ << octal << ” in
octal “;
      return 0;
   }
```

Java Code :

```
//Java program to convert binary number to octal number
import java.util.Scanner;
public class Binary_To_Octal
{
        public static void main(String args[])
        {
                //scanner class object creation
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a binary
number : ");
                int binary = sc.nextInt();
                //Declaring variable to store decimal
number
                int decimal = 0;
                //Declaring variable to use in power

                int n = 0;
                //writing logic for the conversion from
binary to decimal
                while(binary > 0)
                {
                        int temp = binary%10;
                        decimal +=
temp*Math.pow(2, n);
                        binary = binary/10;
                        n++;
                }
                int octal[] = new int[20];
                int i = 0;
                //writing logic for the conversion from
decimal to octal
                while(decimal > 0)
                {
                        int r = decimal % 8;
                        octal[i++] = r;
                        decimal = decimal / 8;
                }
                //printing result
                System.out.print("Octal number : ");
                for(int j = i-1 ; j >= 0 ; j--)
                System.out.print(octal[j]);
                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
#take binary number
Bin_num = 0b10111
#convert using oct() function
Oct_num = oct(Bin_num)
#print number
print('Number after conversion is :' + str(Oct_num))
```

Chapter 31. Decimal to Binary conversion:

The C program to convert decimal numbers into binary numbers is done by counting the number of 1s. The program practices module process and multiplication with base 2 operation for converting decimal into binary number. It further uses modulo operation to check for 1's and hence increases the amount of 1s. The program reads an integer number in decimal in order to change or convert into a binary number.

Ex:- (180)10=(11101000)2

Working:
Step 1: Start
Step 2: Ask the user to insert an integer number in decimal as an input
Step 3: Check whether the entered number is less than or equal to 0
Step 4: Check the divisibility of the number by 2 and store the remainder in the range
Step 5: Increase the range by 1
Step 6: After calculating, print the binary number and the number of 1s.
Step 7: Stop

C Code :

```
/*
 * C program to accept a decimal number and convert it to
binary
 * and count the number of 1's in the binary number
 */


#include<stdio.h>
int main()
{
  //for initialize a variables
   long number, dec_num, rem, base = 1, bin = 0, count =
0;
   //To insert a number
   printf("Insert a decimal num \n");
   scanf("%ld", &number);
   dec_num = number;
   while(number > 0)
   {
      rem = number % 2;
      /*  To count no.of 1s */
      if (rem == 1)
      {
         count++;
      }

      bin = bin + rem * base;
     //number/=2;
      number = number / 2;
      base = base * 10;
   }
  //display
   printf("Input num is = %d\n", dec_num);
   printf("Its binary equivalent is = %ld\n", bin);
   printf("Num of 1's in the binary num is = %d\n", count);
  return 0;
}
```

C++ Code :

```
//C++ Program
   //Decimal to binary conversion
   #include <iostream>
   #include <math.h>
   using namespace std;
   //function to convert decimal to binary
   long convert(int n)
   {
      long binary = 0;
      int i = 1;
      //converting decimal to binary
      while (n!=0)
      {
         int rem = n%2;
         n /= 2;
         binary += rem*i;
         i *= 10;
      }
      return binary;
   }
   //main program
   int main()
   {
      int decimal;
      long binary;
      cout << "Enter a decimal number: ";
      //user input
      cin >> decimal;
      //calling function
      binary = convert(decimal);
      cout << decimal << " in decimal = " << binary << " in
binary" << endl ;
      return 0;
   }
```

Java Code :

```
//Java program to convert decimal number to binary
number
import java.util.Scanner;
public class Decimal_To_Binary
{
        public static void main(String args[])
        {
                //scanner class object creation
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a Decimal
number : ");
                int decimal = sc.nextInt();
                //integer array for storing binary digits
                int binary[] = new int[20];
                int i = 0;
                //writing logic for the conversion
                while(decimal > 0)
                {
                        int r = decimal % 2;
                        binary[i++] = r;
                        decimal = decimal/2;
                }
                //printing result
                System.out.print("Binary number : ");
                for(int j = i-1 ; j >= 0 ; j--)
                System.out.print(binary[j]+" ");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
#take decimal number
dec_num = 124
#convert decimal number to binary
bin_num = bin(dec_num)
#print number
print('Number after conversion is :' + str(bin_num))
```

Chapter 32. Decimal to octal Conversion:

The C program to convert decimal to octal number accepts a decimal number from the user. This number is further converted to its equivalent octal number after following a series of steps. The following section gives an algorithm for this conversion. It is then followed by a C program.

Ex:- If a Decimal number is an octal number we use this method
$(143)_{10}=(217)_8$

Working:
Step 1. Start
Step 2. The user enters a decimal number.
Step 3. Divide the decimal number by 8 to obtain its quotient and remainder. Store remainder in an array.
Step 4. Repeat step 3 with the quotient until the quotient becomes 0.
Step 5. Print the remainder array in reverse order to get the octal conversion
Step 6. Stop

C Code:

```
//Program to convert Decimal number into octal number
#include<stdio.h>

int main()
 {
//Variable initialization
    long dec_num, rem, quotient;
    int i, j, octalno[100];
//Taking input from user
    printf("Enter a number for conversion: ");
//Storing the value in dec_num variable
    scanf("%ld",&dec_num);
    quotient = dec_num;
    i=1;
//Storing the octal value in octalno[] array
    while (quotient!=0)
      {
      octalno[i++]=quotient%8;
      quotient=quotient/8;
      }
//Printing the octalno [] in reverse order
    printf("\nThe Octal of %ld is:\n\n",dec_num);
```

```
  for (j=i-1;j>0;j--)
  //display it
  printf ("%d", octalno[j]);
  return 0;
}
```

C++ Code :

```
//C++ Program
  //decimal to octal conversion
  #include <iostream>
  #include <math.h>
  using namespace std;
  // Function to convert a decimal number to octal
  int convert(int decimal)
  {
    int i = 1, octal = 0;
    //converting decimal to octal
    while (decimal != 0)
    {
      int rem = decimal % 8;
      decimal /= 8;
      octal += rem * i;
      i *= 10;
    }
    return octal;
  }
  //main program
  int main()
  {
    int decimal,octal;
    cout << “Enter a decimal number: “;
    //user input
    cin >> decimal;
    //calling function
    octal = convert(decimal);
    //printing output
    cout << decimal << ” in decimal = “ << octal << ” in
octal”;
    return 0;
  }
```

Java Code :

```
//Java program to convert decimal number to octal number
import java.util.Scanner;
public class Decimal_To_Octal
{
        public static void main(String args[])
        {
                //scanner class object creation
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a Decimal
number : ");
                int decimal = sc.nextInt();
                //integer array for storing octal digits
                int octal[] = new int[20];
                int i = 0;
                //writing logic for the conversion
                while(decimal > 0)
                {
                        int r = decimal % 8;
                        octal[i++] = r;
                        decimal = decimal/8;
                }
                //printing result
                System.out.print("Octal number : ");
                for(int j = i-1 ; j >= 0 ; j--)
                System.out.print(octal[j]);
                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
#take decimal number
Dec_num = 598
#convert using oct() function
Oct_num = oct(Dec_num)
#print number
print('Number after conversion is :' + str(Oct_num))
```

Chapter 33. Octal to Binary conversion :

Octal to binary conversion:-
The C program helps to convert octal numbers to binary numbers. In this program, the user is asked to insert an octal number and by using a loop or if-else statement, the user can convert an octal number to binary number. An integer variable is required to be used in the program.

Working:
Step 1: Start
Step 2: Ask the user to enter an octal number as an input.
Step 3: Store the inserted number in the array octal num.
Step 4: With the help of switch statement, evaluate every number of the octal number
Step 5: Print the equal binary value in a 3 digit number (Eg. 000)
Step 6: Do step 4 under the while loop.
Step 7: Stop

C Code :

```
/*
 * C Program to Convert Octal to Binary number
 */
#include<stdio.h>
#include<conio.h>

#define MAX 1000

int main()
{
    char octalnum[MAX];
    //For initialize
    long i = 0;
    //taking user input of octalnum
    printf("Insert an octal number: ");
    scanf("%s", octalnum);
    printf("Equivalent binary number: ");
    while (octalnum[i])
    {
       //use switch case for multiple condition
        switch (octalnum[i])
        {
          case '0':
               printf("000"); break;
          case '1':
               printf("001"); break;
          case '2':
               printf("010"); break;
          case '3':
               printf("011"); break;
          case '4':
               printf("100"); break;
          case '5':
               printf("101"); break;
          case '6':
               printf("110"); break;
          case '7':
               printf("111"); break;
         //for invalid case
          default:
               printf("\n Invalid octal digit %c ", octalnum[i]);
               return 0;
        }
        i++;
    }
    return 0;
}
```

C++ Code :

```
//C++ Program
    // Octal to Binary conversion
    #include <iostream>
    #include <math.h>
    using namespace std;
    //Function to convert octal to binary
    long convert(int octal)
    {
        int decimal = 0, i = 0;
        long binary = 0;
        //converting octal to decimal
        while(octal != 0)
        {
            int rem = octal%10;
            int res=rem * pow(8,i);
            decimal += res;
            i++;
            octal/=10;
        }
        i = 1;
        //converting decimal to binary
        while (decimal != 0)
        {
            int rem = decimal % 2;
            binary += rem * i;
            decimal /= 2;
            i *= 10;
        }
        return binary;
    }
    //main program
```

```
  int main()
  {
    int octal;
    cout << “Enter an octal number: “;
    //user input
    cin >> octal;
    //function call
    long binary = convert(octal);
    //printing output
    cout << octal << ” in octal = “ << binary << ” in
binary”;
    return 0;
  }
```

Java Code :

```
//Java program to convert octal number to binary number
import java.util.Scanner;
public class Octal_To_Binary
{
        public static void main(String args[])
        {
                //scanner class object creation
                Scanner sc = new Scanner(System.in);
                //input from user
                System.out.print("Enter a octal number
: ");
                int octal = sc.nextInt();
                //Declaring variable to store decimal
number
                int decimal = 0;
                //Declaring variable to use in power

                int n = 0;
                //writing logic for the octal to decimal
conversion
                while(octal > 0)
                {
                        int temp = octal % 10;
                        decimal += temp *
Math.pow(8, n);
                        octal = octal/10;
                        n++;
                }
                int binary[] = new int[20];
                int i = 0;
                //writing logic for the decimal to binary
conversion
                while(decimal > 0)
                {
                        int r = decimal % 2;
                        binary[i++] = r;
                        decimal = decimal/2;
                }
                //printing result
                System.out.print("Binary number : ");
                for(int j = i-1 ; j >= 0 ; j--)
                System.out.print(binary[j]+" ");
                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
#octal number with prefix 0o/0O
oct_num = 0o564
#using bin() to convert octal number to binary
bin_num = bin(oct_num)
#print binary Number
print('Number after conversion is :' + str(bin_num))
```

Chapter 34. Octal to Decimal conversion :

It is easy to convert an octal number to a decimal number. For this, the user is asked to enter an octal number which is converted to a decimal number following a series of steps. The algorithm below illustrates this process in a step wise process. It is then followed by a C program that converts an Octal number to a decimal number.

Working:

Step 1. Start

Step 2. An octal number is taken as an input from the user.

Step 3. Multiply each digit of the octal number starting from the last digit with powers of 8.

Step 4. Add all the digits multiplied.

Step 5. The total sum of the digits gives the decimal number.

Step 6. Stop

C Code :

```
/** C Program to Convert Octal to Decimal */

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

int main()
{
//Variable Initialization
   long int oct, dec = 0;
   int i = 0;
//Taking input from user
   printf("Enter an octal number: ");
   scanf("%ld", &oct);
//Conversion of octal to decimal
   while(oct != 0)
   {
      dec =  dec +(oct % 10)* pow(8, i++);
     //oct/=10;
      oct = oct / 10;
   }
  //display
   printf("Equivalent decimal number: %ld",dec);
   return 0;
}
```

C++ Code :

```
//C++ Program
   //Octal to decimal conversion
   #include <iostream>
   #include <math.h>
   using namespace std;
   // Function to convert octal number to decimal
   int convert(int octal)
   {
      int decimal = 0, i = 0;
      //converting octal to decimal
      while (octal != 0)
      {
         int rem = octal % 10;
         octal /= 10;
         int res=rem*pow(8,i);
         decimal += res;
         i++;
      }
      return decimal;
   }
   //main program
   int main()
   {
      int octal;
      cout << "Enter an octal number: ";
      //user input
      cin >> octal;
      //calling function
      int decimal=convert(octal);
      //printing output
      cout << octal << " in octal = " << decimal << " in
decimal";
      return 0;
   }
```

Java Code :

```
//Java program to convert octal number to decimal number
import java.util.Scanner;
public class Octal_To_Decimal
{
            public static void main(String args[])
            {
                        //scanner class object creation
                        Scanner sc = new Scanner(System.in);
                        //input from user
                        System.out.print("Enter a octal number
: ");
                        int octal = sc.nextInt();
```

```
                    //Declare variable to store decimal
number
                    int decimal = 0;
                    //Declare variable to use in power

                    int n = 0;
                    //writing logic for the conversion
                    while(octal > 0)
                    {
                              int temp = octal % 10;
                              decimal += temp *
Math.pow(8, n);
                              octal = octal/10;
                              n++;
                    }
                    //printing result
                    System.out.println("Decimal number :
"+decimal);
                    //closing scanner class(not compulsory,
but good practice)
                    sc.close();
          }
}
```

Python Code :

```
#take octal number
#with prefix 0o[zero and o/O]
oct_num = 0o512
#convert octal number to integer
#integers are with base 10
deci_num = int(oct_num)
#print number
print('Number after conversion is :' + str(deci_num))
```

Chapter 35. Quadrants in which a given coordinate lies :

The C program reads the coordinate point in a xy coordinate system and identifies the quadrant. The program takes X and Y. On the basis of x and y value, the program will identify on which quadrant the point lies. The program will read the value of x and y variables. If-else condition is used to identify the quadrant of the given value.

Ex:- X and Y coordinates are 20, 30 these lie in 4th quadrant because in mathematics quadrants rules are following

- x is positive integer and y is also positive integer so-that quadrant is a first quadrant.
- x is negative integer and y is positive integer so-that Quadrant is a second quadrant.
- x is negative integer and y is also negative integer so -that Quadrant is a third quadrant.
- x is positive integer and y is negative integer so-that is a first quadrant.

Working:

Step 1: Start
Step 2: The user asked to put value for x and y variables
Step 3: If-else condition is used to determine the value of the given value
Step 4: Check the condition, if x variable’s value is greater than 0 and the variable y is greater than 0.
Step 5: If the condition is true then print the output as the first quadrant.
Step 6: If the condition is false then check the condition if x is lesser than 0 and the y variable is greater than 0.

Step 7: If the condition is true then print the output as a second quadrant.
Step 8: If the condition is false, execute another statement to check if the value of x is less than 0 and y is less than 0.
Step 9: If the condition is true then print the output as the third quadrant.
Step 10: If the condition is false, then check if x variable is greater than 0 and the y value is less than 0.
Step 11: If the condition is true then print the output as the fourth quadrant.
Step 12: If the condition is false, then execute another statement where x value is equal to 0 and y variable is equal to 0.
Step 13: Print the output as an origin.
Step 14: Stop

C Code :

```
#include<stdio.h>
int main()
{
 //for initialization of coordinates
     int x, y; //user input
     printf("Insert the value for variable X and Y\n");
     scanf("%d %d", &x, &y);
 //find true condition of first quadrant
     if (x > 0 && y > 0)
     printf("point (%d, %d) lies in the First quadrant\n",x,y);
 //find second quadrant
     else if (x < 0 && y > 0)
     printf("point (%d, %d) lies in the Second
quadrant\n",x,y);
//To find third quadrant
     else if (x < 0 && y < 0)
     printf("point (%d, %d) lies in the Third
quadrant\n",x,y);
//To find Fourth quadrant
          else if (x > 0 && y < 0)
    printf("point (%d, %d) lies in the Fourth
quandrant\n",x,y);
//To find dose not lie on origin
    else if (x == 0 && y == 0)
    printf("point (%d, %d) lies at the origin\n",x,y);
return 0;
}
```

C++ Code :

```
//C++ program
   //Quadrants in which coordinates lie
   #include<iostream>
   using namespace std;
   //main program
   int main()
   {
     int x, y;
     cout<<"Enter coordinates: \n";
     cin>>x>>y;
     //checking for quadrants and axis
     if(x==0)
        cout<<x<<","<<y<<" lies on y axis";
     else if(y==0)
        cout<<x<<","<<y<<" lies on x axis";
     else if(x>0&&y>0)
        cout<<x<<","<<y<<" lies in 1st quadrant";
     else if(x<0&&y>0)
        cout<<x<<","<<y<<" lies in 2nd quadrant";
     else if(x<0&&y<0)
        cout<<x<<","<<y<<" lies in 3rd quadrant";
     else if(x>0&&y<0)
        cout<<x<<","<<y<<" lies in 4th quadrant";
     else
        cout<<x<<","<<y<<" lies on the origin";
     return 0;
   }
```

Python Code :

```
#take inputs for X and Y
X  = int(input('Enter value for X-axis :'))
Y = int(input('Enter value for Y-axis :'))
#check for 1st quadrant
if X > 0 and Y > 0:
   print('X and Y lie at First quadrant')
#check for 2nd quadrant
elif X < 0 and Y > 0:
   print('X and Y lie at Second quadrant')
#check for 3rd quadrant
elif X < 0 and Y < 0:
   print('X and Y lie at Third quadrant')
#check for fourth quadrant
elif X > 0 and Y < 0:
   print('X and Y lie at Fourth quadrant')else:
   print('X and Y lie at Origin')
```

Chapter 36. Permutations in which n people can occupy r seats in a classroom :

C programming helps in identifying the r number of seats that can be occupied by n number of people. Such a program is known as the possible permutations. Here, We need to write a code to find all the possible permutations in which n people can occupy or number of seats in a classroom/theater.

N students are looking to find r seats in a classroom. Some of the seats are already occupied and only a few can be accommodated in the classroom. The available seats are assumed as r and n number of people are looking to accommodate within the room.

Permutations in which n people can occupy r seats in a classroom in C programming
Way 2 Of Asking Question
Write code to find all possible permutations in which n people can occupy r seats in a theater

Working:
Step 1: Start
Step 2: Ask the user to insert n number of people and the number of seats as r.
Step 3: Calculate permutation, p(n,r).
Step 4: Enter the program to calculate permutation P(n,r) = n! / (n-r)!
Step 5: Print the calculated result.
Step 6: Stop

C Code :

```
#include<stdio.h>

// Program to find the factorial of the number
int factorial (long int x)
{
  long int fact=1,i;
  for(i=1;i<=x;i++)
  {
    fact=fact*i;
  }
  return fact;
}
int main()
{
   long int n,r,permutation,temp;
    long int numerator, denominator;

// Insert the num of people

   printf("\nEnter the number of persons : ");

   scanf("%ld",&n);

// Insert the num of seats

   printf("\nEnter the number of seats available : ");

   scanf("%ld",&r);
// Base condition
// Swap n and r when n is less than r

  if(n < r)
  {
   temp=n;
   n=r;
   r=temp;
  }
  numerator=fact(n);
  denominator=fact(n-r);
  permutation=numerator/denominator;
 printf("\nNum of ways people can be seated : ");
 printf("%ld\n",permutation);
}
```

C++ Code :

```
//C++ Program
   //Permutations in which n people can occupy r seats
   #include<iostream>
   using namespace std;
   //function for factorial
   int factorial(int num)
   {
      int fact=1;
      for(int i=num;i>=1;i–)
         fact*=i;
      return fact;
   }
   //main program
   int main()
   {
      int n,r;
      cout<<"Enter number of people: ";
      //user input
      cin>>n;
      cout<<"Enter number of seats: ";
```

```
        //user input
        cin>>r;
        //if there are more people than seats
        if(r<n)
        {
           cout<<“Cannot adjust “<<n<<” people on “<<r<<”
seats”;
           return 0;
        }
        //finding all possible arrangements of n people on r
seats
        // by using formula of permutation
        int p = factorial(r)/factorial(r–n);
        //printing output
        cout<<“Total arrangements: “<<p;
        return 0;
    }
```

Java Code :

```
import java.util.*;
import java.io.*;

    class PrepInsta
    {
      public static void main(String[] args)
        {
           int n, r, per, fact1, fact2;
           Scanner sc =  new Scanner(System.in);
           System.out.println(“Enter the Value of n and r”);
           n = sc.nextInt();
           r = sc.nextInt();
           fact1 = n;
           for (int i = n – 1; i >= 1; i=i-1)
           {
              fact1 = fact1 * i;  //calculating factorial (n!)
           }
           int number;
           number = n – r;
           fact2 = number;
           for (int i = number – 1; i >= 1; i=i-1)
           {
              fact2 = fact2 * i;  //calculating factorial ((n-r)!)
           }
           per = fact1 / fact2;  //calculating  nPr
           System.out.println(per+“ways”);
    }
}
```

Python Code :

```
#import math lib
import math
#take user inputs
N = int(input('Enter the number of students :'))
R = int(input('Enter the number of seats :'))
#factorial by using factorial() function
nume = math.factorial(N)
deno = math.factorial(N-R)
#permutation = n! / (n-r)!
no_of_ways = nume//deno
#print total no of ways
print('Total number of ways are :' + str(no_of_ways))
```

Chapter 37. Maximum number of handshakes:

In C programming, there’s a program to calculate the number of handshakes. The user is asked to take a number as integer n and find out the possible number of handshakes. For example, if there are n number of people in a meeting and find the possible number of handshakes made by the person who entered the room after all were settled.

Working:
Step 1: Start
Step 2: The user is asked to insert an integer value n, representing the number of people
Step 3: Find nC2, and calculate as n * (n – 1) / 2.
Step 4: Print the outcome derived from the above program
Step 5: Stop

C Code :

```
// C program to find the maximum number of handshakesM
#include<stdio.h>
int main()
{
//fill the code
int num;
scanf("%d",&num);
int total = num * (num-1) / 2; // Combination nC2
printf("%d",total);
return 0;
}
```

C++ Code :

```
//C++ Program
  //Maximum number of handshakes
  #include<iostream>
  using namespace std;
  //main program
  int main()
  {
    int p;
    cout<<“Enter number of Persons: “;
    //user input
    cin>>p;
    cout<<“Maximum number of handshakes: “;
    //find maximum number of handshakes using formula
    int max=p*(p–1)/2;
    //printing output
    cout<<max;
    return 0;
  }
```

Java Code :

```
// Java program to find maximum number of handshakes
import java.io.*;
import java.util.*;

class handshakes
{
  // Calculating the maximum number of handshakes
  static int maxHandshake(int n)
  {
    return (n * (n – 1)) / 2;
  }


  // Driver code
  public static void main (String[] args)
  {
    Scanner sc=newScanner(System.in);
    int n = sc.nextLine();
    System.out.println( maxHandshake(n));
  }
}
```

Python Code :

```
#take user inputs
N = int(input('Enter number of people available :'))
#formula
no_of_handshakes = int(N *((N-1)/2))
#print number of no_of_handshakes
print('Maximum number of handshakes can be :' +
str(no_of_handshakes))
```

Chapter 38. Addition of two fractions:

n this C program we will find sum of two fraction using C

To find the sum of two fractions we will be using the concept of LCM and GCD.

For example: we have to find the sum of 6/2 and 16/3

Firstly the LCM of 2 and 3 will be found. Using the LCM we will convert the numerators i.e. 6 and 16 into digits that can be added and sum of those digits is found, lastly normalization is done using the GCD of sum and LCM.

Working:
Step 1. Start.
Step 2.Initialize variables of numerator and denominator
Step 3. Take user input of two fraction
Step 4. Find numerator using this condition (n1*d2) +(d1*n2) where n1,n2 are numerator and d1 and d2 are denominator .
Step 5. Find denominator using this condition (d1*d2) for lcm.
Step 6. Calculate GCD of this new numerator and denominator .
Step 7. Display a two value of this condition x/gcd,y/gcd);
Step 8. Stop.

C Code :

```
#include <stdio.h>
int main()
{
    //for initialize variables
     int numerator1,
denominator1,numerator2,denominator2,x,y,c,gcd_no;
   //To take user input of numerators and denominators
    printf("\nEnter the numerator for 1st number : ");
    scanf("%d",&numerator1);
    printf("\nEnter the denominator for 1st number : ");
    scanf("%d",&denominator1);
    printf("\nEnter the numerator for 2nd number : ");
    scanf("%d",&numerator2);
    printf("\nEnter the denominator for 2nd number : ");
    scanf("%d",&denominator2);
   //numerator

x=(numerator1*denominator2)+(denominator1*numerator2
);
   //denominator
    y=denominator1*denominator2;
   // Trick part. Reduce it to the simplest form by using gcd.
   for(c=1; c <= x && c <= y; ++c)
   {
      if(x%c==0 && y%c==0)
         gcd_no = c;
   }
   //To display fraction of givien numerators and
denominators
    printf("\nThe added fraction is %d/%d
",x/gcd_no,y/gcd_no);
    printf("\n");
  return 0;
}
```

C++ Code :

```
//C++ Program
   //Adding two fractions
   #include <iostream>
   using namespace std;
   //main Program
   int findGCD(int n1, int n2)
   {
       int gcd;
       for(int i=1; i <= n1 && i <= n2; i++)
       {
          if(n1%i==0 && n2%i==0)
              gcd = i;
       }
       return gcd;
   }
   int main()
   {
       int num1,den1;
       cout << “Enter numerator and denominator of first
number: “;
       //user input
       cin >> num1 >> den1;
       int num2,den2;
       cout << “Enter numerator and denominator of second
number: “;
       //user input
       cin >> num2 >> den2;
       //finding lcm of the denominators
       int lcm = (den1*den2)/findGCD(den1,den2);
       //finding the sum of the numbers
       int sum=(num1*lcm/den1) + (num2*lcm/den2);
       //normalizing numerator and denominator of result
       int num3=sum/findGCD(sum,lcm);
       lcm=lcm/findGCD(sum,lcm);
       //printing output
       cout<<num1<<“/”<<den1<<” +
“<<num2<<“/”<<den2<<” = “<<num3<<“/”<<lcm;
```

```
    return 0;
  }
```

Java Code :

```
//Java program to add two fractions
import java.util.Scanner;
public class add_two_fractions
{
        public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from the user
        System.out.print("Enter numerator for first
fraction : ");
                int num1 = sc.nextInt();
        System.out.print("Enter denominator for first
fraction : ");
                int den1 = sc.nextInt();
        System.out.print("Enter numerator for second
fraction : ");
                int num2 = sc.nextInt();
        System.out.print("Enter denominator for second
fraction : ");
                int den2 = sc.nextInt();
                int num, den, x;
        System.out.print("("+num1+" / "+den1+") +
("+num2+" / "+den2+") = ");
                //logic for calculating sum of two
fractions
                if(den1 == den2)
                {
                        num = num1 + num2 ;
                        den = den1 ;
                }
                else{
                num = (num1*den2) + (num2*den1);
                        den = den1 * den2;
                }
                if(num > den)
                        x = num;
                else
                        x = den;
                for(int i = 1 ; i <= x ; i++)
                {
                        if(num%i == 0 && den%i ==
0)
                        {
                                num = num/i;
                                den = den/i;
                        }
                }
                //logic for getting simplified fraction
                int n = 1;
                int p = num;
                int q = den;
                if( num != den)
                {
                        while(n != 0)
                        {
                        //storing remainder
                        n = num % den;

                        if(n != 0)
                        {
                              num = den;
                                den = n;
                                }
                        }
                }
        System.out.println("("+p/den+" / "+q/den+")");
        //closing scanner class(not compulsory, but good
practice)
                sc.close();

        }
}
```

Python Code :

```
#take inputs
f1_nume = int(input('Enter the numerator for 1st fraction
:'))
f1_deno = int(input('Enter the denominator for the 1st
fraction :'))
f2_nume = int(input('Enter the numerator for 2nd fraction
:'))
f2_deno = int(input('Enter the denominator for the 2nd
fraction :'))
#check if denominators are same
if(f1_deno == f2_deno):
   #add numerator
   Fraction = f1_nume + f2_nume
   #print
   print('Addition of two fractions are :' + str(Fraction) + '/'
+ str(f1_deno))
#if denominators are not same
else:
   #to find the sum
   #denominators should be same
   #apply cross Multiplication
   Fraction = (f1_nume * f2_deno) + (f2_nume * f1_deno)
   print('Addition of fractions are :' + str(Fraction) + '/' +
str(f1_deno * f2_deno))
```

Chapter 39. Replace all 0’s with 1 in a given integer :

The replace all program in C programming works to replace the numbers with zero, where the number must be an integer. All the zeros (if encountered) in the given program will be replaced by 1.

Ex- number is 12004509 all 0’s are replays of 1’s so number is 12114519.

Working:

Step 1: Start
Step 2: The user is asked to insert an integer value as an input
Step 3: Navigate the inserted integer digit by digit
Step 4: If 0 is found, then replace it with 1, and print the integer variable
Step 5: Stop

Question can come like Way 1
Write a code to change all zero's as one's (0s as 1s) in a given number? ex: 120014 needs to be printed as 121114

Question can come like Way 2
implement a c program to replace all 0's with 1 in a given integer as an input, all the 0's in the number has to be replaced with 1.

C Code :

```
// C program to replace all 0’s with 1 in a given integer
#include
int replace (long int num)
{
    // Base case for recursion termination
    if (num == 0)
      return 0;
    // Extract the last digit and change it if needed
    int digit = num % 10;
    if (digit == 0)
       digit = 1;
    // Convert remaining digits and append the last digit

    return replace(num/10) * 10 + digit;
}
int Convert(long int num)
{
    if (num == 0)
      return 1;
    else
     return replace(num);
}
int main()
{
     long int num;
    //To take user input
     printf("\nInsert the num : ");
     scanf("%d", &number);
     //display final result
     printf("\n Num after replacement : %d\n", Convert
(num));
     return 0;
}
```

C++ Code :

```
//C++ Program
   //Convert all 0’s to 1
   #include<iostream>
   using namespace std;
   //main program
   int main()
   {
      int num,num2=0;
      cout<<“Enter number: “;
      //user input
      cin>>num;
      //checking for 0 input
      if(num == 0)
         num2=1;
      //converting 0 to 1
      while(num>0)
      {
         int rem = num%10;
         if(rem == 0)
            rem = 1;
         num = num/10;
         num2=num2*10+rem;
      }
      //converted number
      cout<<“Converted number is: “<<num2;
      return 0;
   }
```

Java Code :

```
//Java program to replace all 0's with 1 in a given integer  :
import java.util.Scanner;
public class replace_0_to_1
{
            public static void main(String[] args)
            {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);
                //input from the user
                System.out.print("Enter the number : 
");
                int number = sc.nextInt();
                //convert the number to string and then 
calculate its length
                String str = Integer.toString(number);
                int len = str.length();
                String str1 = "";
                //use the logic to replace all 0's with 1 
in a given integer
                for(int i = 0 ; i < len ; i++)
                {
                        if(str.charAt(i) == '0')
                                str1 = str1 + '1';
                        else
                                str1 = str1 + 
str.charAt(i);
                }
                System.out.println("Output : "+str1);
                //closing scanner class(not compulsory, 
but good practice)
                sc.close();

        }
}
```

Python Code :

Method 1 :

```
#taking Input
n=int(input())
#converting into string
n=str(n)
#then into the list
n=list(n)
r=” #empty string for addind it the item of list
for i in range(len(n)):
   #if we find ‘0’ we will replace it with ‘1’
   if(n[i]==‘0’):
      n[i]=‘1’
   r=r + n[i]  #creating the new integer
del n
print(int(r))
```

Method 2 :

```
n=int(input(“Enter any number”))
s=str(n)
l=[]
for i in s:
   if(i==‘0’):
      l.append(‘1’)
   else:
      l.append(i)
ns=“”
for i in l:
   ns+=i
print(int(ns))
```

Chapter 40. Can a number be expressed as a sum of two prime numbers :

The program in C takes a positive integer or number which is required to be inserted by the user. The program further identifies whether that digit can be expressed as the sum of two prime numbers. If the inserted number can be expressed as sum of two prime numbers then, print the integer can be expressed as sum of two prime numbers as a result.

Working:

```
Step 1: Start
Step 2: Ask the user to insert a number as an input.
Step 3: Initiate the value of i in a loop from 2 up to half the value of the entered number.
Step 4: Check if i is a prime number.
Step 5: If i is a prime number, identify if (n – 1) is a prime number.
Step 6: If both i and (n – 1) are prime numbers, then the given number can be represented as the sum of prime numbers i and (n – 1).
Step 7: Stop
```

C Code :

```
// C program to check whether a number can be expressed
as a sum of two prime numbers

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

  printf("Insert the num: ");
  scanf("%d", &n);
  int flag = 0;
  for(i = 2; i <= n/2; ++i)
  {
    // Condition for i to be prime
     if(sum_of_two_primes(i) == 1)
     {
       if(sum_of_two_primes(n-i) == 1)
       {
          printf("\n%d can be expressed as the sum of %d
and %d\n\n", n, i, n – i);
          flag = 1;
       }
     }
  }

  if(flag == 0)
     printf("%d cannot be expressed as the sum of two
primes\n", n)

 return 0;
}

   int sum_of_two_primes(int n)
   {
     int i, isPrime = 1;
     for(i = 2; i <= n/2; ++i)
     {
       if(n % i == 0)
       {
         isPrime = 0;
         break;
       }
     }
    return isPrime;
   }
```

C++ Code :

```
//C++ Program
  //Number expressed as sum of two prime numbers
  #include<iostream>
  using namespace std;
  // Function to check prime number
  int Prime(int num)
  {
    int div=0;
    for(int i=1;i<=num;i++)
    {
      if(num%i==0)
        div++;
    }
    if(div==2)
      return 1;
    return 0;
  }
  int main()
  {
    int check = 0, n;
    cout<< "Enter a positive integer: ";
    //user input
    cin>>n;
    for(int i = 1; i <= n/2;i++)
    {
        // condition for i to be a prime number
```

```
            if (Prime(i))
            {
                  // condition for n-i to be a prime number
                  if (Prime(n–i))
                  {
                     cout<<n <<" = "<< i <<" + " << n–i<<
endl;
                     check = 1;
                  }
            }
      }
      if (check == 0)
            cout<<n<<" cannot be expressed as the sum of
two prime numbers.";
      return 0;
   }
```

Java Code :

```
//Java program to check whether a number can be
expressed as a sum of two prime numbers
import java.util.Scanner;
public class number_as_sum_of_two_prime_numbers
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);
                        //input from user
                        System.out.print("Enter a number : ");

                        int number = sc.nextInt();
                        int x = 0;
                        for(int i = 2 ; i <= number/2 ; i++)
                        {
                                    if(prime_or_not(i) == 1)
                                    {

if(prime_or_not(number-i) == 1)
                                                {

System.out.println(number+ " = "+i+" + "+(number-i));
                                                            x = 1;
                                                }
                                    }
                        }
                        if(x == 0)

System.out.println(+number+" cannot be expressed as a
sum of two prime numbers");
            }
      //function for checking number is prime or not
            public static int prime_or_not(int n)
            {
                        int c = 1;
                        for(int i = 2 ; i < n ; i++)
                        {
                                    if(n % i == 0)
                                    {
                                                c = 0;
                                                break;
                                    }
                        }
                        return c;
            }
```

Python Code :

```
#take input
Number = int(input('Enter the Number :'))
#initialize an array
arr = []
#find prime numbers
for i in range(2,Number):
   flag = 0
   for j in range(2,i):
      if i % j == 0:
         flag = 1
   #append prime numbers to array
   if flag == 0:
      arr.append(i)
#possible combinations
flag = 0
for i in range(len(arr)):
   for j in range(i+1,len(arr)):
      #if condition is True Print numbers
      if(arr[i] + arr[j] == Number):
         flag = 1
         print(str(arr[i]) + " and " + str(arr[j]) + ' are prime
numbers when added gives ' + str(Number))
         break
if(flag == 0):
print('No Prime numbers can give sum of ' + str(Number))
```

Chapter 41. Count possible decodings of a given digit sequence :

The decoding programs are the most possible questions asked and are largely practiced in C programming. The program counts the number of possible decodings of the entered digit by the user of the given sequence.
For example, if the digit sequence is “12” then there are two possible decodings for this – One of them is ‘AB’ when we decode 1 as ‘A’ and 2 as ‘B’. Now we can also decode this digit sequence “12” as ‘L’ when we decode digits 1 and 2 taken together as an integer 12
Way 2 of asking Question
Count possible decodings of a given digit sequence -

Working:
Step 1: Start
Step 2: User is required to insert a digit sequence as an input
Step 3: Set count = 0
Step 4: If the last number is not a zero, then return for the next remaining (n-1) numbers and add the results then to the total count.
Step 5: If the last two digits form a valid variable (or smaller than 27), return for the remaining (n-2) numbers and add the outcome to the total calculation.
Step 6: Stop

C Code :

```
//C Program to Count possible decodings of a given digit
sequence
#include<stdio.h>
#include<math.h>

int cnt_decoding_digits(char *dig, int a)
{
   // Initializing an array to store results
    int cnt[a+1];
    cnt[0] = 1;
    cnt[1] = 1;

    for (int k = 2; k <= a; k++) { cnt[k] = 0;
    // If the last digit not equal to 0, then last digit must
added to the number of words if (dig[k-1] > '0')
    cnt[k] = cnt[k-1];

    // In case second last digit is smaller than 2 and last digit
is smaller than 7, then last two digits form a valid character
    if (dig[k-2] == '1' || (dig[k-2] == '2' && dig[k-1] < '7') )
          cnt[k] += cnt[k-2];
 }
 return cnt[a];
}

 int main()
 {
    char dig[15];
    printf("\n Enter the sequence : ");
    gets(dig);
    int a = strlen(dig);
    printf("\n Possible count of decoding of the sequence :
%d\n", cnt_decoding_digits(dig, a));
   return 0;
}
```

C++ Code :

```
//Count possible decodings of a given digit sequence
#include<iostream>
#include<string.h>
using namespace std;
//function to count the number of decodings
int countDecoding(char *digit, int n)
{
            int decodings[n+1];
            decodings[0]=1;
            decodings[1]=1;
            //counting decodings
            for(int i=1;i<=n;i++)
            {
                        int q=digit[i]-48;
                        int p=digit[i-1]-48;
                        if(q>0 && q<=26)
                                    decodings[i+1]=decodings[i];
                        if((q + p*10)>0 && (q + p*10) <=26)
                                    decodings[i+1]
+=decodings[i-1];
            }
            return decodings[n];
}
//main program
int main()
{
            char digit[20];
            cout<<"Input: ";
            //user input
            gets(digit);
            int n = strlen(digit);
            //calling function and printing output
            cout<<"Number of decoding of the sequence
"<<digit<<" are "<<countDecoding(digit,n);
            return 0;
}
```

Chapter 42. Check whether a character is a vowel or consonant :

Given a character, check if it is a vowel or consonant. Vowels are in Uppercase ‘A’, ‘E’, ‘I’, ‘O’, ‘U’ and Lowercase ‘a’, ‘e’, ‘i’, ‘o’, ‘u’. and All other characters both uppercase and lowercase (‘B’, ‘C’, ‘D’, ‘F’,’ b’, ‘c’,’ d’, ‘f’,…..) are consonants. In this article, we will show you, How to write a C program to check Vowel or Consonant with an example.

Working:
We check whether a given character matches any of the 5 vowels. If yes, we print “Vowel”, else we print “Consonant”.

This C program allows the user to enter any character and check whether the user specified character is Vowel or Consonant using If Else Statement.
This program takes the character value(entered by user) as input.
And checks whether that character is a vowel or consonant using if-else statement.
Since a user is allowed to enter an alphabet in lowercase and uppercase, the program checks for both uppercase and lowercase vowels and consonants.
And now we have to follow step’s of C programming

C Code :

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

//To find the corrector is lowercase vowel
isLowerVowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c
== 'u');
//To find the character is Upper case vowel
isUpperVowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c
== 'U');
// compare to charector is Lowercase Vowel or Upper case
Vowel

if (isLowerVowel || isUpperVowel)
printf("%c is a vowel", c);
//to check character is alphabet or not
elseif((c >= 'a' && c= 'A' && c <= 'Z'))
prinf("\n not a alphabet\n");

else
printf("%c is a consonant", c);

return 0;
}
```

C++ Code :

```
//C++ Program to check whether alphabet is vowel or
consonant
#include <iostream>
using namespace std;
//main function
int main()
{
            char c;
            cout<<"Enter an alphabet: ";
            cin>>c;
      //checking for vowels

if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'||c=='A'||c=='E'||c=='I'||c
=='O'||c=='U')
            {
                        cout<<c<<" is a vowel";     //condition
true input is vowel
            }
            else
            {
                        cout<<c<<" is a consonant";
//condition false input is consonant
            }
            return 0;
}
```

Java Code :

```
//JAVA Program to check whether the character entered by
user is Vowel or Consonant.

import java.util.Scanner;
public class vowelorconsonant
{
          //class declaration
            public static void main(String[] args)
{
              //main method declaration
                     Scanner sc=new Scanner(System.in);
//scanner class object creation

                     System.out.println(" Enter a character");
```

```
                        char c = sc.next().charAt(0);
//taking a character c as input from user

                        if(c == 'A' || c == 'E' || c == 'I' || c == 'O' ||
c == 'U'

                        || c == 'a' || c == 'e' || c == 'i' || c == 'o' || c
== 'u')           //condition for the vowels

                                    System.out.println(" Vowel");

                        else if((c >= 'A' && c <= 'Z') || (c >= 'a'
&& c <= 'z'))             //condition for the consonants

                                    System.out.println("
Consonant");
                        else
                                    System.out.println(" Not an
Alphabet");

                        sc.close()       //closing scanner class(not
mandatory but good practice)
            }             //end of main method
}             //end of class
```

Python Code :

```
#get user input
Char =  input()
#Check if the Char belong to set of Vowels
if (Char == 'a' or Char == 'e' or Char == 'i' or Char == 'o' or
Char == 'u'):
   #if true
   print("Character is Vowel")
else:
   #if false
   print("Character is Consonant")
```

Chapter 43. Check whether a character is a alphabet or not :

The C program checks whether the character entered is an alphabet or not. The program makes use of character value inserted by the user. This value can range between lowercase or uppercase alphabets, such as ‘a’ and <= ‘z’ and ‘A’ and <= ’Z’. If the character inserted by the user lies between the above category or ranges then the character is an alphabet and if it does not lie within the given range then it is not.

Working:
Step 1: Start.
Step 2: Insert a character (by the user).
Step 3: Check whether the character lies between a to z and A to Z, if true, print “Alphabet”.
Step 4: condition is false print” not an alphabet”.
Step 5: Stop

C Code :

```
//C Program to find character is alphabet or not
#include<stdio.h>
#include<conio.h>
int main()
{
  char a;

     //Requesting user to insert the character
 printf("Insert any character: ");

    //keeping the inserted character into the variable a
  scanf("%c",&a);

 if( (ch>='a' && ch<='z') || (ch>='A' && ch<=Z'))
    printf("The inserted character %c is an Alphabet”, a);

 else
   printf("The entered character %c is not an Alphabet”, a);
 return 0;
}
```

C++ Code :

```
//Character is Alphabet or not
#include<iostream>
using namespace std;

//main program
int main()
{

char alpha;
cout<<"Enter a character: ";
cin>>alpha;

//checking for alphabet using ASCII value
if((alpha>=65 && alpha<=90)||(alpha>=97 &&
alpha<=122))
{

//input lies in range
cout<<alpha<<" is an alphabet ";
}

else
{
//input does not lie in range
cout<<alpha<<" is not an alphabet ";
}
return 0;
}
```

Java Code :

```
//Java program to check whether the character entered by
the user is an alphabet or not.
import java.util.Scanner;
public class alphabetornot
{
                                        //class declaration
  public static void main(String[] args)
  {
                                        //main method
declaration
      char c;
      Scanner sc = new Scanner(System.in);

          System.out.print("Enter a Character : ");
//Input character
      c = sc.next().charAt(0);

                                        //condition for
checking characters
      if((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
                        System.out.println(c + " is an
Alphabet");
                else
                        System.out.println(c + " is
not an Alphabet");
                sc.close();
//closing scanner class(not compulsory, but good practice)
        }                                               //end
of the main method
}
```

Python Code :

```
n=input()
#ASCII value of the input
x=ord(n)
if(65<=x<=90 or 97<=x<=122):
  print('yes' , n , 'is an Alphabet')
else:
  print('No' , n , 'is not an Alphabet')
```

Chapter 44. Calculate the area of a circle :

I am going to tell you about finding an area of a circle in C programming.The area of a circle is the number of square units inside the circle. Standard formula to calculate the area of a circle is: A=πr² .where π =22/7(value is 3.141)

You can compute the area of a Circle if you know its radius, by simple formula A = 3.14 * r * r in C programming.We allow the user to enter radius, and then find the area of the cirle.

Working:

1. initialization of radius.

2. calculate the area of circle area=3.14*radius*radius.

C Code :

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

//for initialization of radius and area in a float datatype
float radius,area,pi=3.14;

// for use user input

printf("Enter the Radius of a Circle : ");
scanf("%f",&radius);

//formula of area of circle

area = pi*radius*radius;

printf("Area of Circle is: %f",area);
return 0;
}
```

C++ Code :

```
//C++ Program
// area of circle
#include<iostream>
using namespace std;
int main()
{
   float rad;
   cout<<"Enter the radius: ";
   cin>>rad;
   float circleArea = 3.14 * rad * rad;
   cout<<"Area of the circle of radius "<<rad<<"  is
"<<circleArea;
   return 0;
}
```

Java Code :

```
//Java program to calculate area of a circle
import java.util.Scanner;
public class area_of_circle
{
            public static void main(String[] args)
            {
                        //scanner class declaration
                        Scanner sc = new Scanner(System.in);

            //input from the user
                        System.out.print("Enter the radius of a
circle : ");
                        double radius = sc.nextFloat();


           //formula for area of a circle
                        double area = 3.14 * radius * radius;

                        System.out.println(area);

          //closing scanner class(not compulsory, but good
practice)
                        sc.close();
            }
}
```

Python Code :

```
from math import pi
r=float(input("Enter radius of circle :"))
area=pi*r*r
print("The area of circle is",end=" ")
print(area)
from math import pi
d=float(input("Enter diameter of circle :"))
area=(pi*d*d)/4
print("The area of circle is",end="")
print(area)
```

Chapter 45. Find the ASCII value of a character :

ASCII value can be any integer number between 0 and 127 and consists of character variables instead of the character itself in C programming. The value derived after the programming is termed as ASCII value. With the help of casting the character to an integer, the ASCII value can be derived. Every character has an individual ASCII value that can only be an integer. Every time the character is stored into a variable, as a substitute for keeping the character itself, the ASCII value of the specific character will get stored.

Working:

Step 1: Start
Step 2: Ask the user to insert any character
Step 3: The character will be assigned to the variable ‘a’
Step 4: Scan the character variable to find out the ASCII value of the character
Step 5: Stop

C Code :

```
/* C Program to identify ASCII Value of a Character */
#include<stdio.h>
#include<conio.h>
int main()
{
  char a;

  printf("\n Kindly insert any character \n");
  scanf("%c",&a);

  printf("\n The ASCII value of inserted character = %d",a);
  return 0;
}
```

C++ Code :

```
//C++ program to calcualte ASCII value of Character
#include<iostream>
using namespace std;

//main program
int main()
{
        char val;
        cout<<"Enter a character: ";
        cin>>val;

//printing the ASCII value of input
 //through typecasting
cout<<"The ASCII value of "<<val<<" is "<<(int)val;
return 0;
}
```

Java Code :

```
//Java program to print ASCII values of a character

import java.util.Scanner;
 class Main
{
        public static void main(String[] args)
        {
                //scanner class object creation
                Scanner sc=new Scanner(System.in);

                //input from user
                System.out.print("Enter a Character: ");
                char c=sc.next().charAt(0);

                //typecasting from character type to
integer type
                int i = c;

                //printing ASCII value of the character
                System.out.println("ASCII value of
"+c+" is "+i);

                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
#user input
Char = input('Enter the character :')
#convert Char to Ascii value
Asciival = ord(Char)
#print Value
print(Asciival)
```

Chapter 46. Find the prime numbers between 1 to 100 :

We know that the whole number is the basic counting number 0,1,2,3….and so on.So A whole number greater than 1 that can not be made by multiplying other whole numbers.

Example-7 is a prime number because 7 is only divisible by one or itself.It can not be divisible by 2,3,4,5,6.

So a prime number has two factors: 1 and the number itself is called prime numberThe number 1 is neither prime nor composite.

Working:

- First of all we will initialize i,j and count=0 variable.
- the loop is start i=2 to less than equal 100 and second loop start j=1 to less then or equal i.
- the condition is i%j=0.
- and the value of count is incremented after the iteration of the loop.
- and then count values equal to two then print the i. because the prime number has only two factors, one is 1 and another by the number.

C Code :

```
//C Code to print prime number 1 to 100
#include  <stdio.h>



//Main function
  int main()
{
   int i,j,count=0;

//print values between 1 to 100
  printf("The value between 1 to 100\n");

   for(i=2;i<=100;i++)
{
  for(j=1;j<=i;j++)
{
  if(i%j==0)
  count++;
}
if(count==2)
printf("%d",i);
count=0;
}
}
```

C++ Code :

```
//Cpp Code to find prime number between 1 to 100
#include <iostream>
using namespace std;
///Main Function
int main()
{
    int i,j,count=0;

//Print prime number between 1 to 100
cout<<"print prime number between 1 to 100\n";

//For loop for printing values between 1 to 100
for(i=2;i<=100;i++)
{
for(j=1;j<=i;j++)
{
if(i%j==0)
count++;

}
if(count==2)
cout<<" "<<i;
count=0;
}
return 0;
}
```

Java Code :

```
//Java Program to find prime no 1 to 100
public class Main
{
    public static void main(String[] args)
{
   int i,j,count=0;

//print prime no between 1 to 100
System.out.println("prime number between 1 to 100\n");

//loop for printing prime no between 1 to 100
for(i=2;i<=100;i++)
{
for(j=1;j<=i;j++)
{
if(i%j==0)
count++;
}
if(count==2)
System.out.print(" "+i);
count=0;
}
}
```

Python Code :

```
#To find the prime numbers between 1 to 100
li=[] #list of prime numbers will be stored here
for i in range(2,101):
  f=0
  for j in range(2,i+1):
    if(i!=j and i%j==0): #if any n
      f=1
      break

  if(f==0):
    li.append(i)
print('Prime numbers between 1 to 100:')
print(*li,sep=' ')
```

Chapter 47. Calculate the number of digits in an integer :

Today, we will be learning how to code a C program for finding the digits in an integer entered by a user. An integer is made up of a group of digits, i.e; it is a combination of digits from 0-9

Here we will use loops along with an arithmetic operator.This program takes an integer from the user and calculates the number of digits. For example: If the user enters 6589, the output of the program will be 4.
Working:
Step 1: Start
Step 2: The user is asked to insert an integer or number
Step 3: The variable entered by user is stored in variable ‘a’
Step 4: The while loop is iterated till the last expression.
Step 5: After the count, printf “Number of digits”
Step 6: Stop

C Code :

```
#include <stdio.h>

int main()
```

```
{
//to initialize of variable
int count=0;
int n,c;

//to take user input
printf("enter the number: ");
scanf("%d",&n);
c=n;

//when number is zero
if(n==0)
printf("digit is 1");
//false condition

else
{
while(n!=0)
{
//find last digit of number
n=n/10;
//count of a number
++count;
}
printf("the total digit in giving number %d is :
%d",c,count);
}
return 0;
}
```

C++ Code :

```
//C++ Program
//Number of Digits in an Integer

#include<iostream>
using namespace std;

int main()
{

int num,digit=0;
cout<<"Enter any num : ";
 //user input
cin>>num;

//loop to find number digits
do
{
num=num/10;
digit++;
}
while(num!=0);

//output
cout<<"Number of digits in the given integer is: "<<digit;
return 0;
}
```

Java Code :

```
//Java program to find number of digits in an integer
import java.util.Scanner;
public classnumber_of_digits
{
public static void main(String[] args)
        {
                //scanner class declaration
                Scanner sc = new Scanner(System.in);

                //input from user
                System.out.print("Enter an Integer : ");

                int number = sc.nextInt();

                //declare a variable to count number of
digits
                int digit = 0;
                while(number != 0)
                {
                        //pick last digit of the number
and count one by one
                        int pick_last = number % 10;
                        digit++;
                        number = number / 10;
                }

                //display number of digits
                System.out.print("Number of Digits =
"+digit);

                //closing scanner class(not compulsory,
but good practice)
                sc.close();
        }
}
```

Python Code :

```
#get the user input
Integer = int(input('Enter an integer :'))
#initialize the variable
#typecast the integer to string
#store the string
String = str(Integer)
#use len() functiomn to find the length of a String
#print it
print(len(String))
```

Chapter 48. Convert digit/number to words :

The conversion of numbers in words is just a conversion of numeric values to the English format of reading numbers. This code supports the conversion of numbers from 0 – 9999 in English format. Digits have different places when read from its ones place to above. Different places for numbers are:-

- Single digits:- Ones
- Two Digits:-Tens
- Three Digits:- Hundreds
- Four Digits:- Thousands

Working:

- Taking input as a string from the user.
- Check the length of the input.
- if the length is zero print ’empty’ and if the length is greater than 4 print ‘give a string of specific length’
- if length is between 1 – 4, Create arrays for different values.
- Checking the length of the string.
- According to the place of the digit, we will show the output.

C++ Code :

```
//C++ program
  //convert number to text
  #include<iostream>
  #include<string.h>
  using namespace std;
  //main Program
  void numToWords(string num)
  {
    int length_of_string = strlen(num);
    if (length_of_string == 0){
      cout<<“String is Empty”;
      return;
    }

    if (length_of_string > 4){
      cout<<“Please enter the string with supported
length”;
      return;
    }
    string ones_digits = {“zero”, “one”, “two”, “three”,
“four”, “five”, “six”, “seven”, “eight”, “nine”};
    string tens_digits = {“”, “ten”, “eleven”, “twelve”,
“thirteen”, “fourteen”, “fifteen”, “sixteen”, “seventeen”,
“eighteen”,“nineteen”};
    string multiple_of_ten = {“”, “”, “twenty”, “thirty”,
“forty”, “fifty”, “sixty”, “seventy”, “eighty”, “ninety”};
    string power_of_ten = {“hundred”, “thousand”};
    cout<<num<<“:\n“;
    if (length_of_string == 1){
    cout<<ones_digits[num[0] – ‘0’];
    return;
    }
    int x=0;
    while (x < strlen(num)){
      if(length_of_string >= 3){
        if (num[x] – 48 != 0){
          cout<<ones_digits[num[x] – 48]<<“\n“;
          cout<<power_of_ten[length_of_string –
3]<<“\n“;
          length_of_string–;
        }
      }
      else{
        if (num[x] – 48 == 1){
          sum = (num[x] – 48 + num[x] – 48);
          cout<<tens_digits[sum]);
          return;
        }
        else if(num[x] – 48 == 2 and num[x + 1] – 48 ==
0){
          cout<<“twenty”;
          return;
        }
        else{
          int i = num[x] – 48;
          if(i > 0){
            print(multiple_of_ten[i], end = ” “);
          }
          else{
            print(“”, end = “”);
          }
          x += 1;
          if(num[x] – 48 != 0){
            cout<<ones_digits[num[x] – 48];
          }
          }
      }
      x++;
    }
  }
  int main()
  {
    numToWords(“1121”);
    return 0;
  }
```

Python Code :

```
def numToWords(num):
   length_of_string = len(num);
   if (length_of_string == 0):
      print("String is Empty");
      return;
   if (length_of_string > 4):
      print("Please enter the string with supported length");
      return;
   ones_digits = ["zero", "one", "two", "three", "four",
"five", "six", "seven", "eight", "nine"];
   tens_digits = ["", "ten", "eleven", "twelve", "thirteen",
"fourteen", "fifteen", "sixteen", "seventeen",
"eighteen","nineteen"];
   multiple_of_ten = ["", "", "twenty", "thirty", "forty",
"fifty", "sixty", "seventy", "eighty", "ninety"];
   power_of_ten = ["hundred", "thousand"];
   print(num, ":", end = " ");
   if (length_of_string == 1):
      print(ones_digits[ord(num[0]) - '0']);
      return;
   x = 0;
   while (x < len(num)): if (length_of_string >= 3):
         if (ord(num[x]) - 48 != 0):
            print(ones_digits[ord(num[x]) - 48], end = " ");
            print(power_of_ten[length_of_string - 3], end = "
");
         length_of_string -= 1;
      else:
         if (ord(num[x]) - 48 == 1):
            sum = (ord(num[x]) - 48 + ord(num[x]) - 48);
            print(tens_digits[sum]);
            return;
         elif (ord(num[x]) - 48 == 2 and ord(num[x + 1]) -
48 == 0):
            print("twenty");
            return;
         else:
            i = ord(num[x]) - 48;
            if(i > 0):
               print(multiple_of_ten[i], end = " ");
            else:
               print("", end = "");
            x += 1;
            if(ord(num[x]) - 48 != 0):
               print(ones_digits[ord(num[x]) - 48]);
      x += 1;
numToWords("1121")
```

Chapter 49. Counting number of days in a given month of a year:

Number of days in any month of a year can vary specifically in February as the cycle of leap year repeats in every 4 years when the year is leap February gives the count to 29 days but the when the year is not leap it gives count to 28 days and so no of days in a year varies from 365 to 366.
Rather than February every month gives the count of 30 or 31 days in any case whether the year is a leap or not.

Working:

- Take user inputs like month and year.
- Check if the given month is February.
- If True Check if the year is a year leap or not.
- If the year is a leap year Print 29 Days, Else Print 28 Days.
- If Condition in Step 3 is False Check the month.
- Print the number of days assigned to a specific Month.

C++ Code :

```
//C++ program
   //to display number of days in a month
   #include<iostream>
   using namespace std;
   //main Program
   int main()
   {
      //take user inputs for Month and year in integer
      int Month,Year;
      cout<<"\nEnter the Month :";
      cin>>Month;
      cout<<"\nEnter the Year :";
      cin>>Year;
      //Check condition for Month and leap year
      if(Month == 2 && (Year%4 == 0) || ((Year%100 == 0)
&& (Year%400 == 0)))
         cout<<"Number of days is 29";
      else if(Month == 2)
         cout<<"Number of days is 28";
      else if(Month == 1 || Month == 3 || Month == 5 ||
Month == 7 || Month == 8 || Month == 10 || Month == 12)
         cout<<"Number of days is 31";
      else
         cout<<"Number of days is 30";
   }
```

Java Code:

```
import java.io.*;
import java.util.*;
class PrepInsta
{
   public static void main(String args[])
   {
   int month, year;
   Scanner sc = new Scanner(System.in);
   System.out.println("enter the month and year: ");
   month=sc.nextInt();
   year=sc.nextInt();
   if(((month==2) && (year%4==0)) ||
((year%100==0)&&(year%400==0)))
   {
      printf("Number of days is 29");
   }
   else if(month==2)
   {
      printf("Number of days is 28");
   }
   else if(month==1 || month==3 || month==5 || month==7 ||
month==8 || month==10 || month==12)
   {
      printf("Number of days is 31");
   }
   else
   {
      printf("Number of days is 30");
   }
}
}
```

Python Code :

```
#take user inputs for Month and year in integer
Month = int(input('Enter the Month :'))
Year = int(input('Enter the Year :'))
#Check condition for Month and leap year
if(Month == 2 and (Year%4 == 0) or ((Year%100 == 0) and
(Year%400 == 0))):
   #if condition is TRUE
   print('Number of days is 29')
#if False check for other conditions
elif(Month == 2):
   print('Number of days is 28')
elif(Month == 1 or Month == 3 or Month == 5 or Month
== 7 or Month == 8 or Month == 10 or Month == 12):
   print('Number of days is 31')
else:
   print('Number of days is 30')
```

Chapter 50. Finding Number of times x digit occurs in a given input :

In this C++ program we will count the number of occurrences of a given digit in the given input number. The input may lie within the range of integer.

If the digit does not occur in the input it should print 0 else the count of digits.

Sample Input :
 Enter a number : 897982
Enter the digit : 9

Output : 2
 Explanation : The digit 9 occurs twice

Working:

```
1. Start
2. Get the input value from the user.
3. Get the digit from the i/o console.
4. Declare variables n,d,count
n – Given number
d – Digit
count – no. of occurrences
5. Take a while loop.
6. Declare a variable k to store every digit of the number to be compared.
7. Compare k with the digit
if k equals digit increment count.
8. n=n/10
9. Print the value of count.
10. End
```

C++ Code :

```
#include <bits/stdc++.h>

using namespace std;

int main()
{
   //Declare variables

   int n; //given integer
   int d; //given digit
   int count=0; //declare counter variable

   cin >> n >> d; // take input from user
   while(n)
   {

      int k = n%10; // to store the digits of the given input

      n=n/10;

      if(k==d) // compare the given digit with digit of input
      {
         count++; // increment counter variable
      }

   }

   cout << count; // display count of digits

   return 0;
}
```

Python Code :

```
#take user inputs
Number = int(input('Enter the Number :'))
Digit = (int(input('Enter the digit :')))
#initialize Strings
String1 = str()
String2 = str()
#typecast int to str
String1 = str(Number)
String2 = str(Digit)
#count and print the occurrence
#Count function will return int value
#so change it's type to string and concatenate it
print('Digit count is :'str(String1.count(String2)))
```

Chapter 51. Finding number of integers which has exactly x divisors:

Numbers dividing with self or 1 are called prime numbers but numbers having multiple divisors are called composite numbers. In this c++ program, we will find the numbers with the exact number of divisors defined by the user. The divisor of a number is defined as, when we divide a number ‘a’ by other number ‘b’ and gives remainder zero, so the ‘b’ will be considered as the divisor of the ‘a’. We will find the number of the divisor of the numbers and print them along with the count of numbers.

Working:

- Take user inputs like Number and Divisors.
- Initialize a count variable with zero value.
- Run a for loop with a range from 1 to Number+1.
- Initialize another count variable with zero.
- Run other for loop ranging from 1 to iterator of 1st for loop+1.
- Check for complete division conditions and if TRUE increment count2 by 1.
- Come out of for loop and check if count2 is equal to Divisor.
- If TRUE increment count1 by 1 and print the number with exact divisors.
- Print count1.

C++ Code :

```
 //C++ program
   //Strong Number or not
   #include<iostream>
   using namespace std;
   //main Program
   int main()
   {
      int Number,Divisor,count1;
```

```
    cout<<"\nEnter range of number :";
    cin>>Number;
    cout<<"\nEnter exact number of divisors :";
    cin>>Divisor;
    //count1 is to count total number of Numbers with
exact divisor
    count1 = 0;
    for(int i=0;i<=Number;i++)
    {
      //count2 checks the total number of divisors
      int count2 = 0
      //loop to find number of divisors
      for(int j=1;j<=i;j++)
      {
        if(i%j==0)
        {
          count2++;
        }
      }
      if(count2==Divisor)
      {
        count1++;
        cout<<i<<" ";
      }

    }
    cout<<"\n"<<count1;
  }
```

Java Code :

```
import java.io.*;
import java.util.Scanner;
import java.util.*;
public class Prepinsta
{
static int divisors(int num)
{
int count = 0;
for (int i = 1; i <= num; i++)
{
if (num % i == 0)
count = count + 1;
}
return count;
}

static void check(int n)
{
int c = 0;
for (int i = 1; i <= n; i++)
{
if (divisors(i) == 9)
{
System.out.print(i);
System.out.print(" ");
c = c + 1;
}
}
System.out.print("\n\nTotal number of divisors= " + c);

}

public static void main (String[] args)
{
int n;
System.out.print("\nEnter the number of your choice  : ");
Scanner sc = new Scanner(System.in);
n = sc.nextInt();
System.out.print("\n Number which has exactly 9 divisors
are : ");
check(n);
}
}
```

Python Code :

```
#user inputs
Number = int(input('Enter range of number :'))
Divisor = int(input('Enter exact number of divisors :'))
#count1 is to count total number of Numbers with exact
divisor
count1 = 0
#driver loop
for i in range(1,Number+1):
  #count2 checks the total number of divisors
  count2 = 0
  #loop to find number of divisors
  for j in range(1,i+1):
    if i % j == 0:
      count2+=1
    else:
      pass
  if count2 == Divisor:
    count1+=1
    #end = " " is used so it can print Numbers in same line
    print(i,end = " ")
#for break in line between Numbers and total count
print('')
print(count1)
```

Chapter 52. Finding Roots of a quadratic equation :

In this C program, we will find the roots of a quadratic equation [ax2 + bx + c]. We can solve a Quadratic Equation by finding its roots. Mainly roots of the quadratic equation are represented by a parabola in 3 different patterns like :

No Real Roots
One Real Root
Two Real Roots
We get the roots of the equation which satisfies any one of the above conditions :
X = [-b (+or-)[Squareroot(pow(b,2)-4ac)]]/2a

Sample Test Case
Enter value of a :1
Enter value of b :-7
Enter value of c :12
Output
Two Real Roots
4.0
3.0

Working:

Start.
Take input from user a,b,c.
Check the value of a i.e. a!=0.
Calculate Discriminant (D)
*D = b^2 – 4*a*c
If D>0 : Two real roots exist.
If D=0 : Equal root exists.
If D<0 : Imaginary root exists.
Display the existence of roots and the roots of the equation.
End.

C Code :

```
#include <math.h>
#include <stdio.h>
int main() {
   double a, b, c, d, root1, root2, r, i;
   printf("Enter value of a, b and c: ");
   scanf("%lf %lf %lf", &a, &b, &c);

   d = b * b – 4 * a * c;

   // condition for real and different roots
   if (d > 0) {
      printf("Two Real Roots\n");
      root1 = (-b + sqrt(d)) / (2 * a);
      root2 = (-b – sqrt(d)) / (2 * a);
      printf("root1 = %.2lf \nroot2 = %.2lf", root1, root2);
   }

   // condition for real and equal roots
   else if (d == 0) {
      printf("Equal Roots\n");
      root1 = root2 = –b / (2 * a);
      printf("root1 = root2 = %.2lf;", root1);
   }

   // if roots are not real
   else {
      r = –b / (2 * a);
      i = sqrt(-d) / (2 * a);
      printf("No Real Roots\n");
      printf("root1 = %.2lf+%.2lfi \nroot2 = %.2f-%.2fi", r,
i, r, i);
   }

   return 0;
}
```

Java Code :

```
//Java Program to Find the Roots of a Quadratic Equation

import java.util.*;
import java.io.*;

public class PrepInsta {
   // Function to find and display the roots of quadratic
equation
   public static void main(String[] args) {
      Scanner sc= new Scanner(System.in);
      double a,b,c;
         System.out.println("Enter the coefficients of the
quadratic equation");
         a = sc.nextDouble();
         b = sc.nextDouble();
         c = sc.nextDouble();

      double determinant = Math.pow(b,2) – 4*a*c;
      if(determinant > 0){
         System.out.println("Roots are " +
(-b+Math.sqrt(determinant))/(2*a)
                        + " and " +
(-b-Math.sqrt(determinant))/(2*a));
      }else if (determinant == 0){
         System.out.println("Roots are " + -b/(2*a));
      }
      else{
            System.out.println(“Roots are “ + -b/(2*a) + “+i” +
                              Math.sqrt(-determinant)/(2*a) + ” and “
                              + -b/(2*a) + “-i” +
Math.sqrt(-determinant)/(2*a));
        }
    }
}
```

Python Code :

```
#import math library
import math
#take user inputs
a = int(input('Enter value of a :'))
b = int(input('Enter value of b :'))
c = int(input('Enter value of c :'))
#check for value of a
if a == 0:
    print("a cannot be zero")
#if a is greater than 0
else:
    #calculate value of Function
    val = b**2 - 4 * a * c
    root = math.sqrt(abs(val))
    #Check for roots and print according to their nature
    if val > 0:
        print("Two Real Roots")
        print((-b + root)/(2 * a))
        print((-b - root)/(2 * a))
    elif val == 0:
        print("One Real Root")
        print(-b / (2*a))
    else:
        print("No Real Root")
        print(- b / (2*a) , " + i", root)
        print(- b / (2*a) , " - i", root)
```