C - programming

- **Convert days to years, months weeks and days.**

```
#include<stdio.h>
int main()
{
     int days;
     void function(int);
     printf("Enter number of days= ");
     scanf("%d",&days);
     function(days);
}
void function(int days)
       {
             int y,m,w;
               y=days/365;
             days=days%365;
             m=days/30;
             days=days%30;
             w=days/7;
             days=days%7;
             printf("Years=%d,Months=%d,weeks=%d,days=%d",y,m,w,days);
       }
```

C language is both platform-independent and platform-dependent, depending on how it is used.

Platform-independent :- This means that C code can be written on one platform and compiled on another platform.

However, the compiled code of a C program is platform-dependent. This is because the machine code generated by the compiler is specific to the hardware and operating system of the platform on which it is compiled.

Therefore, if you want to run a C program on multiple platforms, you need to compile the source code separately for each platform.

- **Find all palindrome numbers from 1 to n.**

```
#include<stdio.h>
int main()
{
     int n;
     int reverse(int);  //Function declaration
     printf("Enter the value of N=");
     scanf("%d",&n);
     printf("All palindrome numbers are following:-\n");
     for(int i=1; i<=n; i++)
       {
             int rev=reverse(i);  //Function calling
             if(i==rev)
               {
                 printf("%d, ",i);
               }
       }
}
int reverse(int n)  //Function definition
  {
     int rev=0;
     while(n>0)
```

Percentage= (value/total value) X 100

Value= (Percentage/100) X Total value

Swap

C=N1	N1=N1+N2
N1=N2	N2=N1-N2
N2=C	N1=N1-N2

Fahrenheit= 1.8C +32

```
            {
                    rev=rev*10+(n%10);
                    n=n/10;
            }
          return(rev);
       }
```

- **Find prime numbers between two numbers.** **Java programming**

One is not a prime number because it has only one factor, namely 1.

```
import java.util.Scanner;
class Prime
  {
    boolean primeNumbers(int n)
      {
        boolean c=true;
        for(int i=2; i<=(n/2); i++)
          {
            if(n%i==0)
              {
                c=false;
                break;
              }
          }
        return(c);
      }
    public static void main(String args[])
      {
        int s,g;
       System.out.print("Enter the value of N1 and N2= ");
       Scanner scan=new Scanner(System.in);
       Prime obj=new Prime();
       int N1=scan.nextInt();
       int N2=scan.nextInt();
      if(N1<N2)
        {
            s=N1;
            g=N2;
        }
       else
        {
             s=N2;
             g=N1;
        }
       System.out.println("Prime numbers between two numbers are following: ");
       for(int i=s; i<=g; i++)
```

```
        {
          if(obj.primeNumbers(i))
            {
              System.out.print(i+" ");
            }
        }
      }
  }
```

- **Extract even and odd numbers from 1 to n.** **Python programming**

```
def even(n):
    if n%2==0:
        print(n,end=", ")

def odd(n):
    if n%2!=0:
        print(n,end=", ")

N=int(input("Enter the value of N="))
print("Even numbers are following: ")
i=1
while i<=N:
    even(i)
    i+=1

print("\n\nOdd numbers are following: ")
for i in range(N+1):
    odd(i)
```

- **Simple calculator using switch statement** **C programs**

Rectangle Area= Length * Breadth
Rectangle Perimeter= 2*(Length + Breadth)
Circle circumference= 2 * 3.14 Radius
Circle Area= 3.14 * Radius * Radius

```
#include<stdio.h>
int main()
{
            int C,a,b;
            void calc(int,int,int);
            while(true)
              {
                        printf("\n\tOne For Addition\n\tTwo for Subraction\n\tThree for division\n");
                        printf("\tFour for Multiplication\n\tFive for exit\n");
                        printf("Enter your choice=");
                        scanf("%d",&C);
                        if(C==5)
                          {
                          printf("Exit..");
```

```
                    break;
                    }
                  printf("Enter two numbers=");
                    scanf("%d%d",&a,&b);
                  calc(C,a,b);
          }
}

void calc(int C,int a,int b)
  {
          switch(C)
                    {
                            case 1: printf("Addition=%d",a+b);
                                      break;
                            case 2: printf("Subtraction=%d",a-b);
                                       break;
                            case 3: printf("Division=%f",(float)a/(float)b);
                                       break;
                            case 4: printf("Multiplication=%d",a*b);
                                         break;
                              default: printf("Wrong choice\n");
                    }
  }
```

Java programs

- **Print N terms of Armstrong numbers.**

An Armstrong number is a number that is equal to the sum of its own digits raised to the power of the number of digits in the number. For example, 153 is an Armstrong number because 1^3 + 5^3 + 3^3 = 153.

```
import java.util.Scanner;
class Armstrong
 {
  boolean checkArmstrong(int n, int digits)
    {
      double sum=0,num=n,d;
      while(n>0)
        {
         d=n%10;
         sum=sum+Math.pow(d,digits);
         n=n/10;
        }
        if(num==sum)
         {
         return true;
         }
        else
        {
         return false;
```

```
       }
    }
  public static void main(String s[])
    {
      Scanner scan=new Scanner(System.in);
      System.out.print("How many armstrong numbers do you want to print= ");
      int N=scan.nextInt();
      Armstrong obj=new Armstrong();
      for(int count=1,i=1; count<=N; i++)
        {
          int digits=0,n=i;
          while(n>0)
           {
            digits=digits+1;
            n=n/10;
           }
          if(obj.checkArmstrong(i,digits))
           {
            System.out.print(i+" ");
            count++;
           }
        }
    }
 }
```

- **Find greatest number from 3 numbers using Conditional/ternary operator.**

```
#include<stdio.h>
int main()
  {
            int a,b,c,G;
            printf("Enter three numbers=");
            scanf("%d%d%d",&a,&b,&c);
            G=(a>b)? ((a>c)? a:c):((b>c)? b:c);
            printf("Greatest number= %d",G);
  }
```

- **Find greatest and smallest digits of a number.** **C programs**

```
#include<stdio.h>
int main()
  {
            long N,G,S;
            printf("Enter a number=");
            scanf("%d",&N);
            G=N%10;
```

```
        S=N%10;
        while(N>0)
          {
                int d=N%10;
                if(d>G)
                 {
                        G=d;
                 }
                if(d<S)
                 {
                        S=d;
                 }
                 N=N/10;
          }
        printf("Greatest digit= %d\t Smallest digit=%d",G,S);
  }
```

- **Print 1st digit and last digit of a number.**

```
#include<stdio.h>
int main()
  {
        long N,FD,LD;
        printf("Enter a number=");
        scanf("%d",&N);
        LD=N%10;
        while(N>0)
          {
           FD=N%10;
           N=N/10;
          }
        printf("1st digit= %d \t Last digit=%d",FD,LD);
  }
```

ASCII American Standard Code for Information Interchange

A to Z =	65 to 90
A to z =	97 to 122
0 to 9 =	48 to 57

Simple interest = (principle * time * rate) / 100

Compound Interest = [{P * (1 + r/100)^t}-P]

Where,

P is principal amount, R is the rate and

T is the time

- **C program for Convert a string character lower to upper case and upper to lower case.**

```
#include<stdio.h>
int main()
 {
        char str[12];
        int i;
        printf("Enter a string value=");
        scanf("%s",&str);
        printf("Entered string is: %s",str);
```

Functions for string character converter.

#include <ctype.h>

Ch=tolower(ch);

Ch=toupper(ch);

```
          for(i=0; str[i]!='\0'; i++)
           {
                    if(str[i]>=65 && str[i]<=90)
                      {
                              str[i]=str[i]+32;
                      }
                    else
                      {
                              str[i]=str[i]-32;
                      }
           }
           printf("\nCase converted string is: %s",str);
           return(0);
 }
```

- **Enter two number, Check 1st number is divisible by 2nd number or not, if not then print before and next number that is divisible by entered 2nd number.**

```
import java.util.Scanner;
class Div
{
   int[] divNumbers(int n1,int n2)
    {
      int Num[]=new int[2];
      int remainder=n1%n2;
      Num[0]=n1-remainder;
      Num[1]=n1+(n2-remainder);
      return(Num);
    }
    public static void main(String args[])
     {
      Scanner scan=new Scanner(System.in);
      System.out.print("Enter 1st and 2nd number= ");
      int n1=scan.nextInt();
      int n2=scan.nextInt();
      if(n1%n2==0)
       {
          System.out.println("Entered 1st number is divisible by 2nd number");
       }
      else
       {
          Div obj=new Div();
          int n[]=obj.divNumbers(n1,n2);
          System.out.println("Entered 1st number is not divisible by 2nd number");
```

A triangle is valid if the sum of all the three angles is equal to 180 degrees and if the sum of any two sides of a triangle is greater than the third side.

Area of a triangle with 3 sides: √**[s(s-a)(s-b)(s-c)]** where

Semi perimeter(S) = (a + b + c)/2 **OR**

Area = 1/2 ×(base × height)

```
            System.out.println("Before and after "+n1+" respectively-> "+n[0]+" and "+n[1]+" is divisible by "+n2);
        }
      }
}
```

- **Print table of from 1 to N in python.**

```
def table(n):
    i=1
    while i<=10:
        t=i*n
        print(n,"*",i,"=",t)
        i+=1

N=int(input("Enter the value of N="))
for n in range(1,N+1,1):
    table(n)
    print()
```

- **Print all divisors of a number in python.**

```
def divisors(n):
    i=1
    while(i<=(n//2)):
        if n%i==0:
            print(i,end=",  ")
        i+=1
    print(n)


num=int(input("Enter a number="))
print("All divisors are following of ",num,":")
divisiors(num)
```

- **Implementation of class and object in python.**

```
class Summantion:       # Class declaration
    def __init__(self,a,b):    #Constructor
        self.x=a
        self.y=b
    def sum(self):      #Function
        s=self.x + self.y
        return s        #Function returns value

obj1=Summantion(7,8)   #Object creation
obj2=Summantion(11,12)
```

```
print("Sum of 7 and 8= ",obj1.sum())    #Function calling and print result
print("Sum of 11 and 12= ",obj2.sum())
```

- **Check a number is prime or not in C programming language.**

```
# include<stdio.h>
int main()
{
        int n;
        int prime(int);
        printf("Enter a number= ");
        scanf("%d",&n);
        if(prime(n))
          {
                printf("%d is a prime number",n);
          }
        else
          {
                printf("%d is not a prime number",n);
          }
}

  int prime(int n)
    {
        int P=1,i=2;
        while(i<=(n/2))
          {
                if(n%i==0)
                  {
                        P=0;
                        break;
                          }
                          i+=1;
                  }
                return(P);
        }
```

Note:

One is not a prime number.

Two is a prime number.

- **Check a number is Perfect number or not in C.**

```
# include<stdio.h>
int main()
{
        int n;
        void perfect(int);
        printf("Enter a number= ");
```

A perfect number is a positive integer that is equal to the sum of its positive divisors, excluding the number itself.

For example, 6 has divisors 1, 2 and 3, and 1 + 2 + 3 = 6, so 6 is a perfect number.

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


 void perfect(int n)
       {
        int sum=0, i=1;
        while(i<=(n/2))
          {
                if(n%i==0)
                 {
                        sum+=i;
                  }
                 i+=1;
            }

                        if( sum==n)
                          {
                                printf("%d is a perfect number",n);
                          }
                        else
                          {
                                printf("%d is not a perfect number",n);
                          }
        }
```

- **Display following series of Nth terms in C. 2,5,10,17….n 2,10,26….n 4,16,36…..n 5,17,37……n**

```
#include<stdio.h>
int main()
 {
        int c,n;
        void series1(int);
        void series2(int);
        void series3(int);
        void series4(int);
        while(true)
          {
                printf("1 for 2,5,10,17...Nth \n2 for 2,10,26...Nth\n3 for 4,16,36...Nth\n");
                printf("4 for 5,17,37...Nth,\n5 for exit\n");
                printf("\nEnter your choice= ");
                scanf("%d",&c);
                if(c==5)
            {
                printf("\nExit...");
            break;
                 }
                printf("\nEnter the number of terms= ");
          scanf("%d",&n);
                if(c==1)
                 {
                      series1(n);
                 }
                else if(c==2)
                 {
                      series2(n);
                 }
                else if(c==3)
                 {
                      series3(n);
                 }
                else if(c==4)
                 {
                      series4(n);
                 }
                else
                 {
                      printf("\nYour choice is wrong try again..\n");
                 }

         }
}

    void series1(int N)
     {
        int i=1,v;
        while(i<=N)
         {
               v=i*i+1;
               printf("%d, ",v);
               i+=1;
          }
        printf("\n\n");
 }

void series2(int N)
```

```
  {
        int i=1,v,terms=1;
        while(terms<=N)
         {
                v=i*i+1;
                printf("%d, ",v);
                i+=2;
                terms+=1;
         }
        printf("\n\n");
  }

        void series3(int N)
         {
                int i=2,v,terms=1;
                while(terms<=N)
                 {
                        v=i*i;
                        printf("%d, ",v);
                        i+=2;
                        terms+=1;
                 }
                printf("\n\n");
         }

      void series4(int N)
         {
                int i=2,v,terms=1;
                while(terms<=N)
                 {
                        v=i*i+1;
                        printf("%d, ",v);
                        i+=2;
                        terms+=1;
                 }
                printf("\n\n");
         }
```

- **Check a year is leap year or not in C.**

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

```
If ((Y%100==0 && Y%400==0) || (Y%100!=0 && Y%4==0))
                printf("Leap year")
    else
                printf("Not leap year");
```

```
        void leapYear(int);
        printf("Enter a year= ");
        scanf("%d",&y);
        leapYear(y);
 }

  void leapYear(int Y)
   {
        if(Y%100==0)
         {
                        if(Y%400==0)
                         {
                               printf("%d is a leap year",Y);
                         }
                       else
                         {
                               printf("%d is not a leap year",Y);
                         }
                }
               else
                {
                       if(Y%4==0)
                 {
                       printf("%d is a leap year",Y);
                         }
                       else
                         {
                               printf("%d is not a leap year",Y);
                         }
                }
        }
```

- **Print Nth terms of Fibonacci series in java.**

```
import java.util.Scanner;
class Fib
 {
   static void fibonacci (int terms)
     {
       int a=0,b=1,c=0,i=1;
       while(i<=terms)
         {
          System.out.print(c+",  ");
          a=b; b=c; c=a+b;
          i+=1;
```

The Fibonacci series is the sequence of numbers, where every number is the sum of the preceding two numbers, such that the first two terms are '0' and '1'.

Then a Fibonacci series can thus be given as,

0, 1, 1, 2, 3, 5, 8, 13, 21, 34…….N.

```
        }
      }
    public static void main(String args[])
      {
        Scanner scan=new Scanner(System.in);
        System.out.print("Enter the number of terms= ");
        int t=scan.nextInt();
        fibonacci(t);
      }
 }
```

- **Print reverse of a number also print sum of all digits of entered number in java.**

```
import java.util.Scanner;
class Rev
{
    int reverse(int n)
     {
        int rev=0, sum=0;
        while(n>0)
          {
            int remainder=n%10;
            sum=sum + remainder;
            rev=rev*10 + remainder;
            n=n/10;
          }
        System.out.println("Reverse of enteted number= "+rev);
        return(sum);
     }
    public static void main(String args[])
      {
        Rev obj=new Rev();
        System.out.print("Enter a number= ");
        Scanner scan=new Scanner(System.in);
        int n=scan.nextInt();
        int s=obj.reverse(n);
        System.out.print("Sum of all digits of enteted number= "+s);
      }
}
```

- **Find power of a number without using power function in java.**

```
import java.util.Scanner;
class Pow
 {
    static int powValue(int x,int y)
    {
      int v=1;
      while(y>0)
       {
         v=v*x;
         y--;
       }
      return(v);
    }
  public static void main(String args[])
    {
      int a,b,p;
      Scanner scan=new Scanner(System.in);
      System.out.println("Enter two numbers= ");
      a=scan.nextInt();
      b=scan.nextInt();
      p=powValue(a,b);
      System.out.println("Power value= "+p);
    }
 }
```

- **Print all prime numbers between 1 to N in java.**

```
import java.util.Scanner;
class Prime
 {
   static void primes(int N)
    {
      int P,i,n;
      for(n=2; n<=N; n++)
        {
          P=1;
          i=2;
              while(i<=(n/2))
                {
                    if(n%i==0)
                      {
                          P=0;
                         break;
                      }
                  i++;
                }
             if(P==1)
               {
                 System.out.print(n+",  ");
              }
        }
    }
   public static void main(String args[])
    {
      Scanner scan=new Scanner(System.in);
      System.out.print("Enter the value of N= ");
      int N=scan.nextInt();
      System.out.println("All prime numbers are following between 1 to "+N+": ");
      primes(N);
    }
 }
```

- **String implementation in C.**

```
#include<stdio.h>
#include<string.h>
int main()
{
        char str1[]="String implementaion";
        char str2[]={'H','E','L','L','O','\0'};
        char str3[5]="Abcd";
        char str4[5]={'w','x','y','z','\0'};
        printf("Enterd string are following:\n\n");
        printf("%s\n%s\n%s\n%s\n",str1,str2,str3,str4);

        //String functions
        printf("\n%d\n%s\n%s\n%s",strlen(str1),strrev(str2),strupr(str3),strlwr(str2));
}
```

String functions in C.

strcat()

strcmp()

strcpy()

strrev()

- **Sorting of ten names in C.**

```
#include<stdio.h>
#include<string.h>
int main()
{
        char str[5][11];
        void strSort(char[][11]);
        printf("Enter five students names=\n");
        for(int i=0; i<5; i++)
          {
                printf("Enter %dth student name= ",i+1);
                scanf("%s",str[i]);
          }
        printf("\nEntered students names are following:\n");
        for(int i=0; i<5; i++)
```

Note:

When reading a string with scanf, we don't need to use the & operator. The name of the array itself acts as a pointer to the first element of the array, which is what scanf needs.

```
            {
                printf("[%d] %s\n",i+1,str[i]);
             }
        printf("\nSorted students names are following:\n");
        strSort(str);
}

void strSort(char str[][11])
   {
        int i,j, len; char s[11];

        for(i=0; i<5; i++)
          {
                for(j=i+1; j<5; j++)
                  {
                        if(strcmp(str[i],str[j])>0)
                         {
                                strcpy(s,str[i]);
                                strcpy(str[i],str[j]);
                                strcpy(str[j],s);
                         }
                  }
           }
        for(int i=0; i<5; i++)
              {
                  printf("[%d] %s\n",i+1,str[i]);
              }
   }
```

- **Find reverse of a string value without using string function in C.**

```
#include<stdio.h>
char* strRev(char[]);
int main()
{
        char str[11];
        printf("Enter a string value= ");
        scanf("%s",str);
        char* sr= strRev(str);
        printf("\nReverse of entered string: %s",sr);
        return 0;
}


  char* strRev(char str[])
```

```
   {
          int i,j; char s;
          for(i=0; str[i]!='\0'; i++);

          for(i--,j=0; j<i; i--,j++)
            {
                    s=str[i];
                    str[i]=str[j];
                    str[j]=s;
                      }
          return(str);
          }
```

- **Check a string value is palindrome or not without using string function in C.**

```
#include<stdio.h>
int strPlin(char[]);
int main()
{
          char str[11];
          printf("Enter a string value= ");
          scanf("%s",str);
          int P= strPlin(str);
          if(P==1)
            {
                    printf("Entered string is palindrome");
            }
          else
            {
                    printf("Entered string not is palindrome");
            }
          return 0;
}

 int strPlin(char str[])
   {
          int i, j, len,P=1;

   // Find the length of the string
   for(len=0; str[len]!='\0'; len++);

   // Check if the string is a palindrome
   for (i = 0, j = len - 1; i <= j; i++, j--)
            {
                      if (str[i] != str[j])
                    {
                         P=0;
                         break;
                      }
            }
      return P;
   }
```

- **Check a string value is palindrome or not without using string function in C.**

```
#include<stdio.h>
#include<string.h>
int main()
{
         char str1[11],str2[11];
         printf("Enter two string values= ");
         gets(str1);
         gets(str2);
         if(strcmp(str1,str2)==0)
           {
                  printf("Entered string values are same\n");
                  puts(str1); puts(str2);
           }
         else
           {
                  printf("Entered string values are not same\n");
                  puts(str1); puts(str2);
           }
         return 0;
}
```

- **Check a entered number is fibonacci number or not in java.**

```
import java.util.Scanner;
class Fibonacci
 {
   void fibNum(long n)
     {
       long a=0, b=1,c=0; int P=0;
      while(c<=n)
        {
         if(c==n)
          {
            P=1;
            break;
          }
```

```
           a=b;   b=c;   c=a+b;
         }
        if(P==1)
         {
           System.out.println("Entered number is a fibonacci number");
         }
        else
         {
           System.out.println("Entered number is not a fibonacci number");
         }
      }

    public static void main(String args[])
      {
        long num;
        System.out.println("Entered number a number");
        Scanner scan=new Scanner(System.in);
        num=scan.nextLong();
        Fibonacci obj=new Fibonacci();
        obj.fibNum(num);
      }
 }
```

Recursion is the technique of making a function call itself.

This technique provides a way to break complicated problems down into simple problems which are easier to solve.

An array in C is a collection of elements of the same data type, stored in a contiguous block of memory.

Each element in an array is identified by an index, which is a non-negative integer that represents its position in the array. The first element in an array has an index of 0, and the last element has an index of n-1, where n is the size of the array.

Note: In C, integer arrays are not terminated with a specific character like null ('\0') character. Therefore, the loop continues until it reaches the end of the array, which is determined by the size of the array.

But, In C, character arrays (strings) are terminated with a null character ('\0'), If you declare a char array of size 5 in C, it can store up to 4 characters plus a null character ('\0') at the end. If you try to store more than 4 characters in the array, you may encounter issues such as buffer overflow or data corruption.

- **Find factorial of a number using recursion in C.**

```
#include<stdio.h>
int main()
{
        int n,F;
        int factorial(int);
        printf("Enter a number=");
        scanf("%d",&n);
        F=factorial(n);
        printf("Factorial of %d= %d",n,F);
}

 int factorial(int n)
   {
                if(n==0)
                 {
                        return 1;
                 }
                else
                 {
                        return (n*factorial(n-1));
```

```
                    }
    }
```

- **Find smallest and greatest elements from an array list in C.**

```
#include<stdio.h>
int main()
{
        void greatest_smallest(int[]);
        int Ar[7],i;
        printf("Enter 7 numbers in array list=\n");
        for(i=0; i<7; i++)
          {
                printf("Enter %dth number= ",i+1);
                scanf("%d",&Ar[i]);
          }
        printf("Entered array elements are following: \n");
        for(i=0; i<7; i++)
         {
                printf("%d, ",Ar[i]);
         }
        greatest_smallest(Ar);
        return(0);

}
  void greatest_smallest(int Ar[])
    {
        int i,len,G,S;
        for(len=0; Ar[len]!='\0'; len++);
          G=S=Ar[0];
         for(i=0; i<len; i++)
               {
                        if(G<Ar[i])
                             {
                                 G=Ar[i];
                          }
                         if(S>Ar[i])
                          {
                                 S=Ar[i];
                          }
              }
        printf("\nGreatest element= %d\tSmallest element= %d",G,S);
      }
```

$$\text{Number of array elements} = \frac{\text{Size of whole array}}{\text{Size of 1 array element}}$$

```
int len=sizeof(Ar) / sizeof(Ar[0]);
```

- **Arrange an array elements in ascending order.**

```
#include<stdio.h>
int main()
{
            int* ascending(int[],int);
            int Ar[7],i, *As;
            printf("Enter 7 numbers in array list=\n");
            for(i=0; i<7; i++)
              {
                        printf("Enter %dth number= ",i+1);
                        scanf("%d",&Ar[i]);
              }
            printf("Entered array elements are following: \n");
            for(i=0; i<7; i++)
             {
                        printf("%d, ",Ar[i]);
             }

            As=ascending(Ar,7);
            printf("\nSorted array elements are following:\n");
                for(i=0; i<7; i++)
                  {
                        printf("%d, ",As[i]);
                  }
            return(0);
}

  int* ascending(int Ar[],int len)
    {
       int i,j,z;
          for(i=0; i<len; i++)
            {
                         for(j=i+1; j<len; j++)
                   {
                        if(Ar[i]>Ar[j])
                          {
                            z=Ar[i];          Ar[i]=Ar[j];           Ar[j]=z;
                          }
                    }
             }
              return(Ar);
}
```

- **Swap two numbers using pointer in c language.**

```
#include<stdio.h>
int main()
{
        void swap(int*, int*);
        int a,b;
        printf("Enter 1st number=");
        scanf("%d",&a);
        printf("Enter enter 2nd number= ");
        scanf("%d",&b);
        printf("Entered 1st and 2nd numbers are following:\n");
        printf("1st= %d\t2nd= %d",a,b);
        swap(&a,&b);
        printf("\n\nSwaped 1st and 2nd numbers are following:\n");
        printf("1st= %d\t2nd= %d",a,b);
}
  void swap(int *x, int *y)
   {
        int z;
        z=*x;   *x=*y;   *y=z;
   }
```

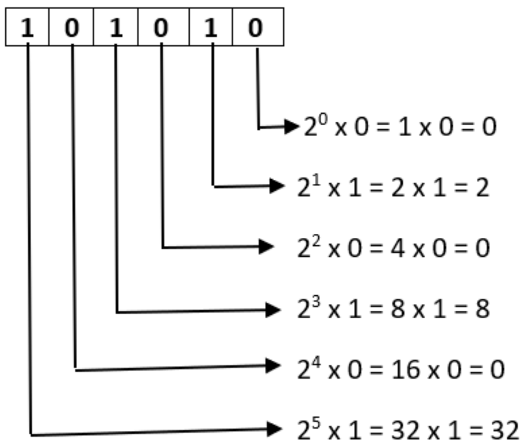
A pointer is a variable that stores the memory address of another variable as its value.

Pointers allow us to manipulate memory directly and efficiently, and to create more complex data structures such as linked lists, trees, and graphs.

- **Find factorial from1 to n in c.**

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

 int factorial(int n)
   {
        int fact=1;
        while(n>0)
          {
                fact=fact*n;
```

Convert binary number to decimal number.


1 0 1 0 1 0
$2^0 \times 0 = 1 \times 0 = 0$
$2^1 \times 1 = 2 \times 1 = 2$
$2^2 \times 0 = 4 \times 0 = 0$
$2^3 \times 1 = 8 \times 1 = 8$
$2^4 \times 0 = 16 \times 0 = 0$
$2^5 \times 1 = 32 \times 1 = 32$

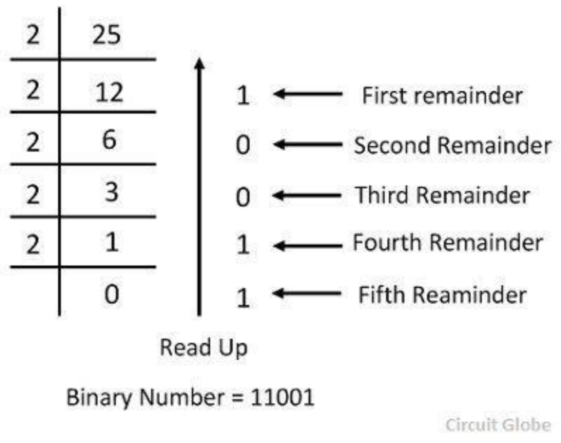
Resultant decimal number= 0+2+0+8+0+32 = **42**

```
                n--;
            }
          return(fact);
   }
```

- **Convert binary number to decimal number in c.**

```
#include<stdio.h>
#include<math.h>
int main()
{
          int binToDec(int);
          int bin,n,r,P=1,dn;
          printf("Enter a binary number=");
          scanf("%d",&bin);
          n=bin;
          while(n>0)
            {
                    r=n%10;
                    if(r!=0  &&  r!=1)
                      {
                              P=0;
                              break;
                      }
                    n=n/10;
            }
          if(P==1)
            {
                    dn=binToDec(bin);
                    printf("Decimal number= %d",dn);
            }
          else
            {
                    printf("Enter only binary number");
            }
}
 int binToDec(int bin)
   {
          int r,sum=0,p=0;
          while(bin>0)
            {
                    r=bin%10;
                    sum=sum+(pow(2,p)* r);
                    bin=bin/10;
                    p++;
```

Convert decimal number to binary number.



Recursion function to convert binary to decimal.

```
  int binToDec(int bin)
      {
              if(bin==0)
                {       return(0);     }
                else
                  {

          return((bin%10)+2*binToDec(bin/10));
                  }
      }
```

```
            }
          return(sum);
   }
```

- **Search a number in an array list in c.**

```
#include<stdio.h>
int main()
{
          void search(int[],int);
          int Ar[5],sn,i;
          printf("Enter 5 numbers=\n");
          for(i=0; i<5; i++)
            {
                    scanf("%d",&Ar[i]);
            }
          printf("\nEnter a number for search=");
          scanf("%d",&sn);
          search(Ar,sn);
}
  void search(int Ar[],int sn)
    {
          int P=0,i;
          for(i=0; Ar[i]!='\0'; i++)
            {
                    if(sn==Ar[i])
                      {
                              P++;
                                }
                      }
          if(P>0)
            {
                    printf("\n%d is avaliable %d times in array list.",sn,P);
                      }
                    else
                     {
                              printf("\n%d is not avaliable in array list.",sn);
                     }
          }
```

- **Find greatest and smallest number from various numbers without using array in c.**

```
#include <stdio.h>
int main()
{
    int n, num, max, min, i;
    printf("How many numbers do you want to enter..? ");
    scanf("%d", &n);
    printf("Enter 1st number:");
    scanf("%d", &num);
    min= max = num;
    for (i = 1; i < n; i++)
      {
                printf("Enter %d number:",i+1);
        scanf("%d", &num);
        if (num > max)
          {
            max = num;
           }
        if (num < min)
           {
           min = num;
           }
        }
    printf("\nThe greatest number is: %d\n", max);
    printf("The smallest number is: %d\n", min);
    return 0;
}
```

- **Check a char is upper case, lower case, number and special char in c.**

```
#include<stdio.h>
int main()
{
            char c;
            printf("Enter a character= ");
            scanf("%c",&c);
            if(c>=65 && c<=90)
              {
                        printf("\nEntered character is upper case");
              }
            else if(c>=48 && c<=57)
              {
                      printf("\nEntered character is number");
              }
            else if(c>=97 && c<=122)
              {
                        printf("\nEntered character is lower case");
              }
            else
              {
                    printf("Entered character is special character");
            }
return(0);
}
```

- **Search a char from a sentence with their position in c.**

```
#include<stdio.h>
int main()
{
          void charSearch(char[],char);
          char sen[23],c;
          printf("Enter a sentance= ");
          fgets(sen,sizeof(sen),stdin);
          printf("Enter a char for search= ");
          scanf("%c",&c);
          charSearch(sen,c);
}

  void charSearch(char s[],char c)
    {
          int i,P=1;
          for(i=0; s[i]!='\0'; i++)
            {
                    if(c==s[i])
                      { P=0;
                              printf("\n%c is avaliable %dth position in the entered sentence",c,i+1);
                      }
            }
                    if(P==1)
                      {
                              printf("\n%c is not avaliable any position in the entered sentence",c);
                      }
   }
```

fgets() is a standard library function in C that is used to read a line of text from a file stream or from the standard input (stdin) and store it in a character array (string). It is safer than the gets() function because it takes a limit argument that specifies the maximum number of characters to read, preventing buffer overflow.

- **Find greatest from 3 numbers using pointer in c.**

```
#include<stdio.h>
int main()
{
          void greatest(int*,int*,int*,int*);
          int x,y,z,G;
          printf("Enter three numbers= ");
          scanf("%d%d%d",&x,&y,&z);
          greatest(&x,&y,&z,&G);
            printf("Greatest number= %d",G);
}
  void greatest(int *a,int *b,int *c,int *g)
    {
                    *g=*a;
                     if(*b>*g)
                        {
                                *g=*b;
                       }
                      if(*c>*g)
                       {
                              *g=*c;
                       }
   }
```

- **Print records of five students in c using structure.**

```
#include<stdio.h>
struct students{
  char name[23],SRN[12];
  struct dob{
           int d,m,y;
  }D;
  long int mob;
};

int main()
{
  struct students s[5];
  int i;
  printf("Enter records of 5 students= \n");
  for(i=0; i<5; i++)
    {
              printf("Enter name,SRN,mobile number and DOB of %d student= ",i+1);
             scanf("%s%s%ld",s[i].name,s[i].SRN,&s[i].mob);
             scanf("%d%d%d",&s[i].D.d,&s[i].D.m,&s[i].D.y);
     }
  printf("Entered records of five students are following:\n");
  printf("\nSRN\tName\tDOB\t\tMobile\n");
        for(i=0; i<5; i++)
            {
           printf("%s\t%s\t%d-%d-%d\t%ld\n",s[i].SRN,s[i].name,s[i].D.d,s[i].D.m,s[i].D.y,s[i].mob);
           }
}
```

In C, a structure is a collection of different data types (such as int, float, char, etc.) that are grouped together under a single name. It is a user-defined data type that allows us to combine various types of data into a single entity.

- **Print records of a student using union in c.**

```
#include<stdio.h>
union student{
  char name[23],SRN[12];
  long int mob;
};

int main()
{
  union student s;
  printf("Enter name of student= ");
  scanf("%s",s.name);
  printf("Entered student name is: %s",s.name);
  printf("\n\nEnter SRN of student= ");
  scanf("%s",s.SRN);
  printf("Entered student SRN is: %s",s.SRN);
  printf("\n\nEnter mobile number of student= ");
  scanf("%ld",&s.mob);
  printf("Entered student mobile number is: %ld",s.mob);
  return(0);
}
```

A union is a user-defined data type that allows us to store different data types in the same memory location.

When a union variable is declared, the compiler allocates memory for the largest member of the union, and all members share this same memory space.

Only one member of the union can be accessed at a time, and modifying one member will overwrite the data in the shared memory location, affecting all members.

Exploring Java's Special

- **Write a Java programme to count the number of vowels, consonants, and spaces in a string value without using any of Java's predefined functions.**

```
import java.util.Scanner;
class Counter
  {
    int[] vowelConsonantCounter(String str)
      {
        int vowels=0, consonants=0, spaces=0, val[]=new int[3];

        for(int i=0; i<str.length(); i++)
          {
            char c=str.charAt(i);
            if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'||c=='A'||c=='E'||c=='I'||c=='O'||c=='U')
              {
                vowels++;
              }
            else if(c==' ')
              {
                spaces++;
              }
            else
              {
                consonants++;
              }
          }
          val[0]=vowels;
          val[1]=consonants;
          val[2]=spaces;
        return(val);
      }
    public static void main(String args[])
      {
        String sen=new String();
        System.out.print("Enter a string values= ");
        Scanner scan=new Scanner(System.in);
        sen=scan.nextLine();
        Counter obj=new Counter();
        int V[]=obj.vowelConsonantCounter(sen);
        System.out.println("Number of vowels= "+V[0]);
        System.out.println("Number of consonants= "+V[1]);
        System.out.print("Number of spaces= "+V[2]);
      }
  }
```

- **Write a Java programme to print a pyramid structure.**

```
import java.util.Scanner;
class Pyramid
 {
   void pyramidPrint(int rows)
     {
       int spaces=rows-1, stars=1;
       for(int i=1; i<=rows; i++)
        {
          for(int j=1; j<=spaces; j++)
            {
              System.out.print(" ");
            }
          for(int k=1; k<=stars; k++)
            {
              System.out.print("*");
            }
          System.out.println();
          spaces--;
          stars+=2;
        }
      }

        public static void main(String args[])
         {
            System.out.print("Enter number of rows= ");
            Scanner scan=new Scanner(System.in);
            int R=scan.nextInt();
            Pyramid obj=new Pyramid();
            obj.pyramidPrint(R);
         }
 }
```

- **Write a Java programme to create an Array list, initialise it with five objects, and print using the forEach loop.**

```
import java.util.ArrayList;
class ArrayListA
{
    public static void main(String[] args)
     {
        ArrayList<String> myList = new ArrayList<>();
          // Initializing ArrayList with 5 objects
          myList.add("Apple");
          myList.add("Banana");
          myList.add("Orange");
          myList.add("Grapes");
          myList.add("Mango");
          for (String fruit : myList)
              {
                 System.out.println(fruit);
              }
     }
}
```

- **Find the reverse of a string without using any functions.**

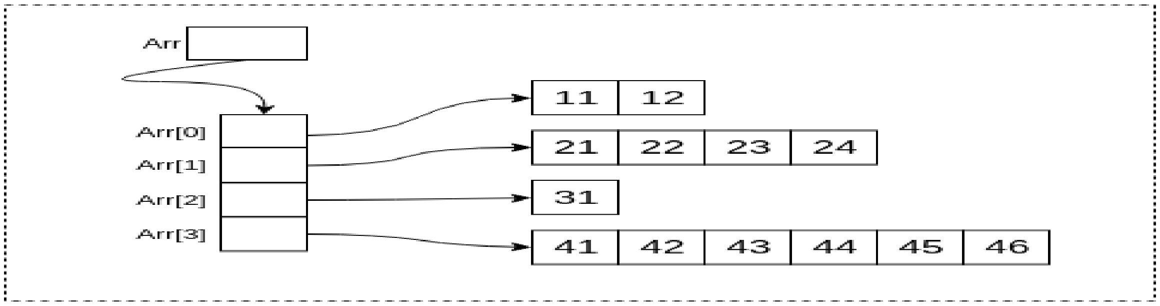
```
import java.util.Scanner;
class StringReverse
  {
    static String stringReverse(String str)
      {
        String s="";   //OR String s=new String();
        for(int i=str.length()-1;    i>=0;    i--)
           {
            s=s+str.charAt(i);
        }

        return s;
    }
    public static void main(String s[])
     {
        System.out.print("Enter a string value=");
        Scanner scan=new Scanner(System.in);
        String str=scan.nextLine();
        System.out.printf("Entered string is= %s",str);
        System.out.println("\nReverse of entered string= "+stringReverse(str));
     }
 }
```

- **Convert string characters from lower to upper and upper to lower case without using any string function.**

```
import java.util.Scanner;
class Strings
 {
   String stringCaseCoverter(String S)
    {
     String s="";
     for(int i=0; i<S.length(); i++)
       {
        char ch=S.charAt(i);
        if(ch>=65 && ch<=90)
          {
           s=s+ (char)(ch+32);
          }

        if(ch>=97 && ch<=122)
          {
           s=s+(char)(ch-32);
          }
       }
       return s;
    }

    public static void main(String s[])
     {
        System.out.print("Enter a string value=");
        Scanner scan=new Scanner(System.in);
        String Str=scan.nextLine();
        Strings obj=new Strings();
```

```
            System.out.print("Coverted string= "+obj.stringCaseCoverter(Str));
          }

      }
```

- **A jagged array is an array whose elements are arrays. In a jagged array, the number of rows is fixed, but the number of columns is not. It is also called an array of arrays.**



```
import java.util.Scanner;
class Jagged
  {
    int[][] jaggedArrayInput(int N)
      {
        int Ar[][]=new int[N][]; /*{{1,4,5,6},
                                  {1,3,4,5,6,7},
                                  {3,4,5,6,7}};
        Ar[0]=new int[]{1,2,3,4,5,6};
        Ar[1]=new int[]{2,3,4,6};*/
        Scanner scan=new Scanner(System.in);
        for(int i=0; i<N; i++)
          {
            System.out.print("How many elements do you want to insert in "+(i+1)+"th row=");
            int n=scan.nextInt();
            Ar[i]=new int[n];
            for(int j=0; j<Ar[i].length; j++)
              {
                System.out.print("Enter "+(j+1)+"th element= ");
                Ar[i][j]=scan.nextInt();
              }
          }
          return Ar;
      }
     void jaggedArrayOutput(int Ar[][])
      {
        System.out.println("Entered elements in jagged array are following=");
        for(int i=0; i<Ar.length; i++)
          {
            for(int j=0; j<Ar[i].length; j++)
```

```
              {
                System.out.print(Ar[i][j]+"  ");
              }
              System.out.println();
          }
      }

    public static void main(String src[])
      {
        System.out.print("How many rows do you want to create in a jaggad array=");
        Scanner scan=new Scanner(System.in);
        int N=scan.nextInt();
        Jagged obj=new Jagged();
        int Ar[][]=obj.jaggedArrayInput(N);
        obj.jaggedArrayOutput(Ar);
      }
  }
```

- **Check a entered string is palindrome or not.**

```
import java.util.Scanner;
class Palindrome
 {
    void palindromeCheck(String s)
      {
       int P=1;
        for(int i=0, j=s.length()-1;  i<=j; i++,j--)
          {
            if(s.charAt(i)!=s.charAt(j))
              {
                P=0;
                break;
              }
          }
        if(P==1)
          {
            System.out.println("Entered string is palindrome");
          }
        else
          {
            System.out.println("Entered string is not palindrome");
          }
      }

    public static void main(String src[])
      {
        System.out.print("Enter a string value= ");
        Scanner scan=new Scanner(System.in);
        String s=scan.nextLine();
        Palindrome obj=new Palindrome();
        obj.palindromeCheck(s);
      }
 }
```

- In Java, the Object class is the root class of all classes. Every class in Java, either directly or indirectly, inherits from the Object class.

- In Java, association is a relationship between two or more classes that establishes a connection or dependency between them. It represents how objects of one class are related to objects of another class.

॥ॐ नमः शिवाय॥

|| ॐ तत्पुरुषाय विद्महे, महादेवाय धीमहि, तन्नो रूद्र प्रचोदयात् ||

♨ಌ★ଓ♨

<u>Thank you</u>