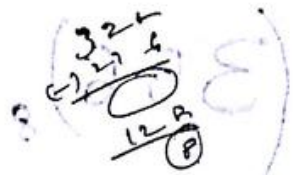
28/4/16



322 → Decimal

Binary →

512	256	128	64	32	16	8	4	2	1
1	1	0	0	0	1	0	0	0	

$(322)_{10} \rightarrow (101000100)_2$

octal → Take Binary num and group it to 3

1 1 0 0 0 1 0 0 0

4 2 1 | 4 2 1 | 4 2 1

6 1 0

Why group it to 3?
octal num from 0 to 7

$(322)_{10} \rightarrow (101000100)_2 \rightarrow (610)_8$

Hexadecimal → Take Binary Equallent Num & group it to 4

0 0 0 1 1 0 0 0 1 0 0 0 $\Rightarrow (188)_{16}$

8 4 2 1 | 8 4 2 1 | 8 4 2 1

1 8 8

$(322)_{10} \rightarrow (101000100)_2 \rightarrow (188)_{16}$

$(356)_8$

Binary →

3 5 6

4 2 1 4 2 1 4 2 1

$(011\ 101\ 110)_2$

$(356)_8 \rightarrow (011101110)_2$

Decimal →

0 1 1 1 0 1 1 1 0

256 128 64 32 16 8 4 2 1

$(238)_{10}$

128 + 64 + 32 + 8 + 4 + 2 = 238

$(356)_8 \rightarrow (238)_{10}$

Hexa decimal →

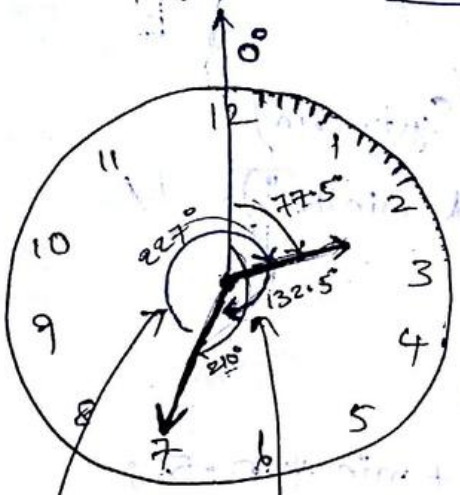
1 1 1 0 1 1 1 0 → $(EE)_{16}$

E E

$(356)_8 \rightarrow (011101110)_2 \rightarrow (EE)_{16}$

$(AB)_{16}$

(*) WAP to find the angle b/n hour Arm & minute Arm in Analog clock. hr 2 : 35 min



highest angle

least angle

Total ← 360°

$\frac{360}{12} \Rightarrow 30°$

Hour hand Every hour → Rotate 30°

Hour hand Every min → moves 0.5°

60 min → 30°

min → 0.5

$\frac{30}{60} = \frac{1}{2} = 0.5$

(*) Take Every time least angle

2 hr 35 min

$\Rightarrow 2 \times 30° + 35 \times 0.5$

$\Rightarrow 60° + 17.5°$

Hour hand moves → 77.5°

For Every min hour hand moves 0.5° with minute hand

35 min

$35 \times 6 \Rightarrow 210$

⇒ minute hand moves

1 min ⇒ 6°

$\frac{360}{60} \Rightarrow 6°$

1 min ⇒ $\frac{360}{60} \Rightarrow 6°$

~~77.5 × 210 ⇒ 132.5~~

210
77.5
132.5

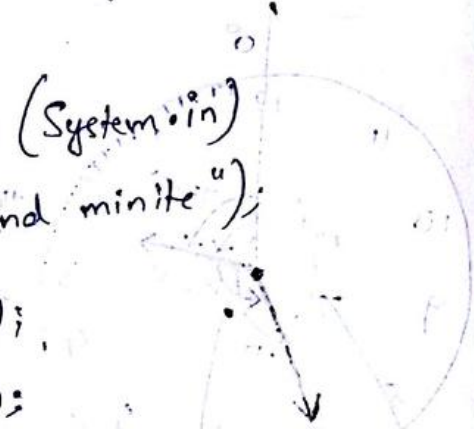
least angle ⇒ 132.5

132.5 < 180°

360
132.5
227.5 ⇒ highest angle

227.5 > 180°

```
class Clock
{
  psvm( )
  {
    Scanner sc = new Scanner (System.in)
    SOP ("Enter hour and minite");
    int hr = sc.nextInt();
    int min = sc.nextInt();
    double hangle = hr * 30 + min * 0.5;
    double mangle = min * 6;
    double angle = 0.0;
    if (hangle > mangle)
      angle = hangle - mangle;
    else
      angle = mangle - hangle
    if (angle > 180°)
      angle = 360 - angle;
    SOP ("last angle :" +angle);
  }
}
```



(*) WAP to print the multiplication table of given num

```
int n=3;
for (i=n; i<=n; i++)
    for (j=1; j<=10; j++)
        SOP ("i + " * " + J + " = " + (i*j));
            n +                       (n*j));
```

possible ways to write

```
for( ; cond ; Exp);
for( ; ; ; ) ;  ⟶ Infinite loop (Defaultly condition is true)
for( ; ; Exp) ;
for(Ini ; cond ; ) ;
```

Note :- In output stmt SOP

+ is highest priority than *

(*) WAP to display Sanju if num is multiple of 3 else display geeta if num is multiple of 5 or display Sanju weds geeta if it is multiple of 3 & 5

```
n = sc.nextInt();
if (n%3==0 && n%5==0)
  SOP("Sanju weds geeta");
else if (n%5==0)
  SOP("geetha");
else if (n%3==0)
  SOP("Sanju");
```

29/4/16

(*) Factorial of given Num

```
n = Sc.nextInt();
int fact = 1;
while (n != 0)  || while (n >= 1)
{
   fact = fact * n;
   n--;
}
Sop("fact of num is" + fact);
```

(*) factorial of given 2 numbers

```
class main()
{
static int fact (int n)
   {
   int fact = 1
      while (n >= 1)
      {
       fact = fact * n;
        n--;
      }
   return fact;
}
class Mainclass
{ psvm()
  {
   for (int i=1; i<=10; i++)
    Sop("factorial of" + i + "is" + fact(i));
}}
```

1 - 10

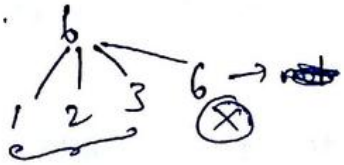
```
class main()
{ int fact (int n)
   { int fact = 1;
     while (n >= 1)
     { fact = fact * n;
     }.n--;
     return fact;
}
class mainclass
{
 psvm()
 {..
```

Given num is perfect or not

⇓

Sum of its divisors is Equal to that num.

[∵ The number itself also a one divisor. But we will not consider that one]



1+2+3 ⇒ 6

```
class Perfect
{
Static int Perf (int n)
   { int perf=0;
     for(i=1; i<=n/2; i++)
     { if (n%i==0)
       {
         perf = perf + i
       }
     }
   return perf;
}
import java.util.Scanner;
class Mainclass
{
  PSVM( )
   { Scanner sc = new Scanner(System.in);
     n = sc.nextInt();
int i = Perfect.perf(n);
   if (i==n);
   {
     Sop ("n+"is a perfect num");
   }
   else
   Sop("n+"is not perfect num");
}
```

2/9/16

Prime

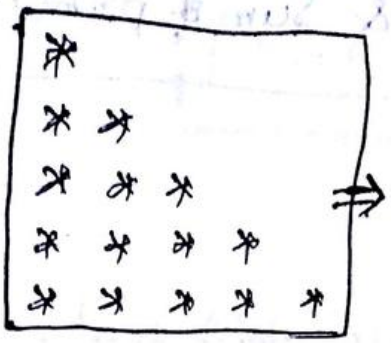
Given num is prime or not,

```
class Mainclass
{
static int isprime (int n)      || static boolean isprime(int n)
{
    int temp = 0;
    for (int i=2 ; i<=n/2 ; i++)
    {
        if(n % i == 0)
        {
            temp = i;
            break;
        }
    }
    return temp;
}
Public static void main (String[] args)
{
    SOP("-----");
    int p = isprime(5);          || boolean p = isprime(5);
    if(p==0)                     || if(isprime(5));
    SOP("num is prime num");  →
    else                      →
    SOP("num is not a prime");  →
}
```

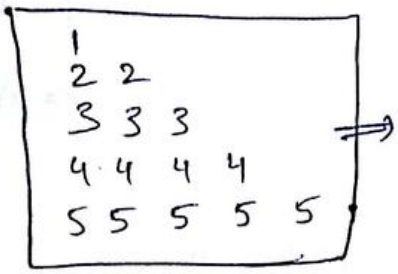
```
if (temp==0)
    return true;
else
    return false;
```

Prime → 0 to 100 & count & Sum of prime no's

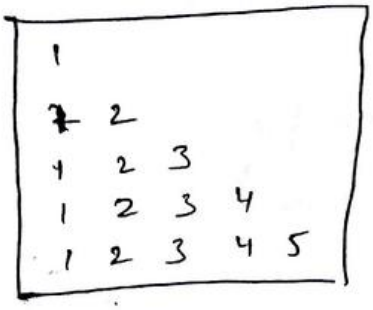
```
class Mainclass
{
 static boolean isprime (int n)
 {
   int temp = 0;
   for (int i=2 ; i<=n/2 ; i++)
   {
     if (n%i == 0)
     {
       temp=1;
       break;
     }
   }
   if (temp==0)
   return true;
   else
   return false;
 }
 Public static void main(String[] args)
 {
   Sop("----"); int Sum=0; int count=0;
   for(int i=2; i<=100 ; i++)
   {
     if(isprime(i))
     {
       Sum = Sum +i;
       count++;
     } Sop(i);
   }
   Sop("sum of prime no's b/n 0-100 is" :+Sum);
   Sop("no of prime no's b/n 0-100 is" + count);
}
```



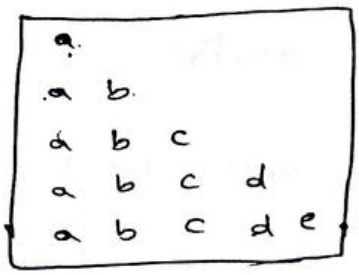
```
for (int i=1 ; i<=n ; i++)
  for (int j=1; j<=i ; j++)
     Sop("* ");
  Sopln();
```



```
for (int i=1 ; i<=n; i++)
  for (int j=1; j<=i ; j++)
    Sop(i);
  Sopln();
```

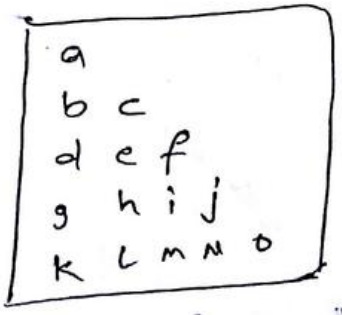


```
for (int i=1 ; i<=n; i++)
  for (int j=1; j<=i ; j++)
```

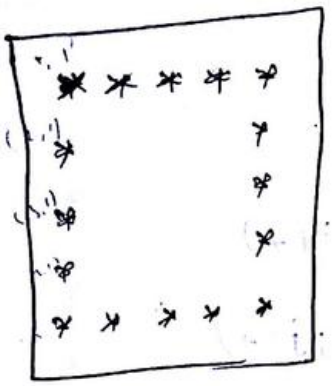


```
char ch;
for(int i=1 ; i<=n ; i++)
{                              → ch='a';
  for (int j=1 ; j<=i ; j++)
  {
    ~~char ch='a';~~
        Sop (ch++); // → Sop((char)ch++);
  }
} Sopln();
```

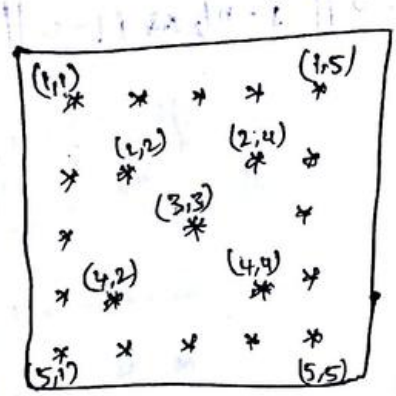
ch++ ⇓ it prints Ascii values



```
char ch='a';
for (int i=1; i<=n; i++)
  for(int j=1; j<=i ; j++)
    Sop((char)ch++);
  Sopln();
```



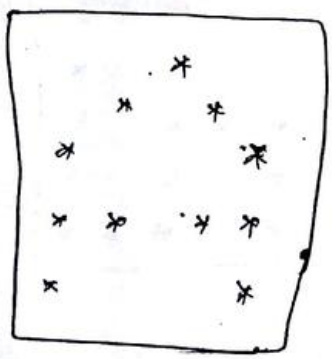
```
for (i=1; i<=5; i++)
  for (j=1; j<=5; j++)
  {
    if (i==1 || j==1 || i==5 || j==5)
      Sop("* ");
  }
```



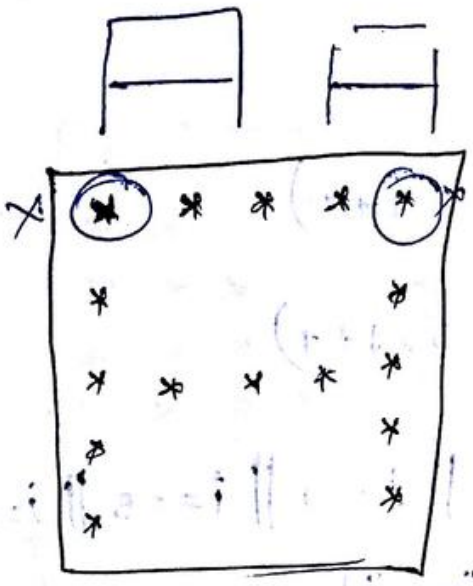

```
for (i=1; i<=5; i++)
{
  for (j=1; j<=5; j++)
  {
    if (i==j || i+j==6 || i==1 || j==1 ||
        i==5 || j==5)
      Sop("* ");
  }
  Sopln();
}
```

(or)

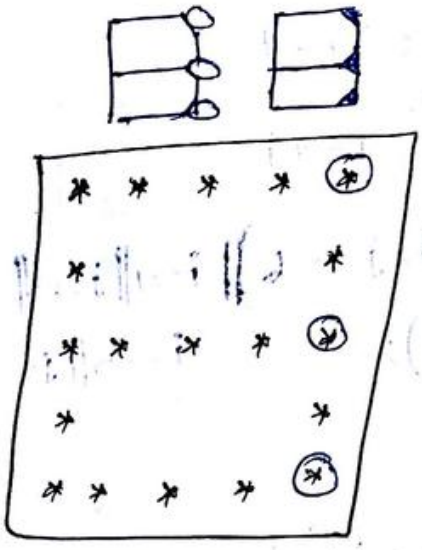
```
for (i=1; i<=n; i++)
  for (j=1; j<=n; j++)
    if (i==1 || j==1 || i==5 || j==5 || i==j || i+j==n+1)
      Sop("* ");
  Sopln();
```



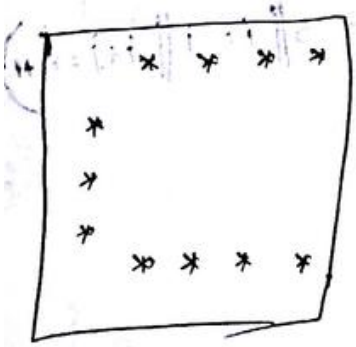
```
for (i=1; i<=5; i++)
  for (j=1; j<=5; j++)
    if (i==4 || (i==1 && j==3) || (i==2 && j==2) ||
        (i==2 && j==4) || (i==3 && j==1) || (i==3 && j==5)
        || (i==5 && j==3) || (i==5 && j==5))
      Sop("* ");
```



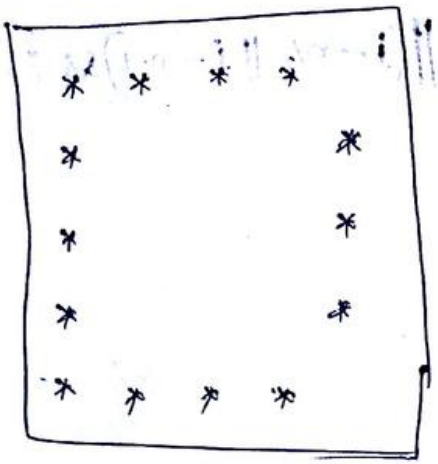
```
int n=13
for(int i=0 ; i<n ; i++)
 for (int j=0 ; j<=n/2 ; j++)
  if (i==0 && j!=0 && j!=n/2 ||
      j==0 && i!=0 || j=n/2 && i!=0 ||
        i==n/2)
   Sop("* ");
```



```
int n=13;
for (int i=0; i<n; i++)
 for (j=0; j<=n/2 ; j++)
  if(((i==0 || i==n/2 || i==n-1) &&
```

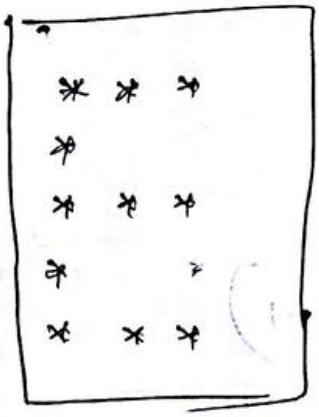


```
if(((i==0 || i==n-1) && j!=0 || j=0 && i!=0 &&
                                  i!=n-1))
   Sop ("*  ");
  else
  Sop("  ");
```



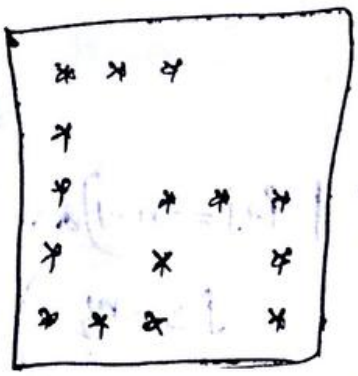
```
if((i==0 || i==n-1) && j!=n/2 || j==0 ||
    j==n/2 && i!=0 && i!=n-1)
```

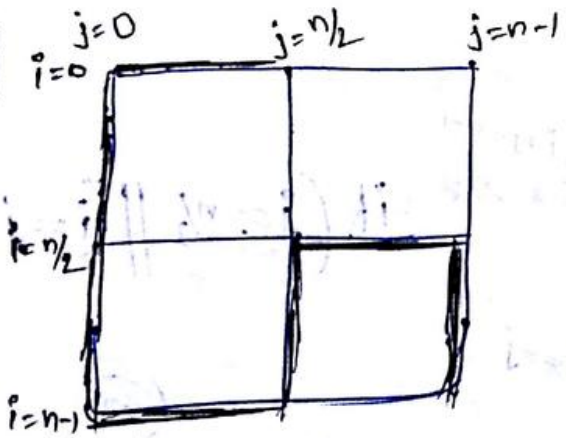
3	5	16



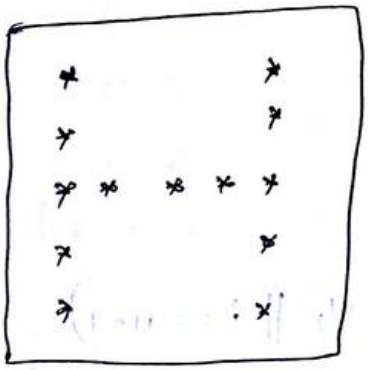
```
if( i==0 || j==0 || i==n/2 || i==n-1)
```

```
if( i==0 || j==0 || i==n/2 ).
```

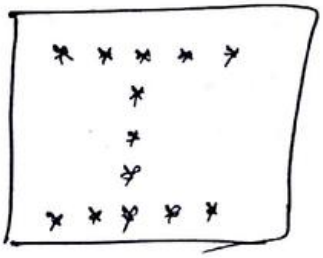




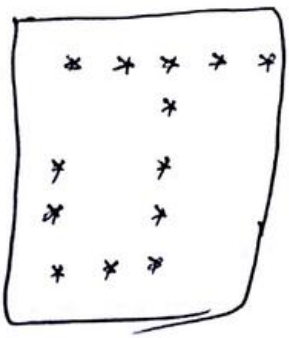
```
if (j==0 || (i==0 || i==n-1) && j<=n/2 || (j==n/2 || j==n-1) && i>=n/2
                || (i==n/2 && j>=n/2)
Sop("*");
Soplm();
```



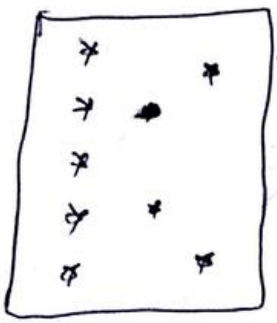
```
if (j==0 || i==n/2 || j==n-1)
```



```
if (i==0 || j==n/2 || i==n-1)
```



```
if (i==0 || j==n/2 || (i==n-1 && i<=n/2) ||
    (j==0 && j>=n/2))
```

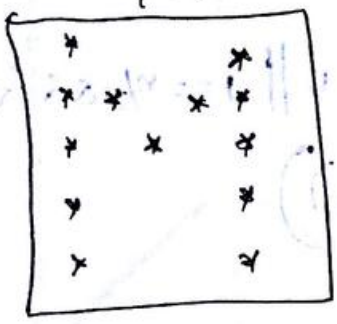



```
if (j==n/2 || (i==j || i+j==n-1) &&
    j>=n/2)
```

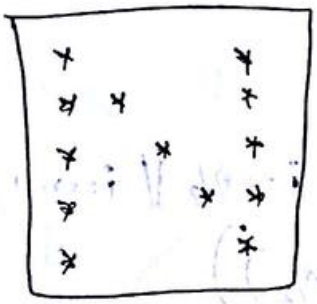
(67)

```
if (j==0 || (i==j || i+j==n-1) &&
    j<=n/2)
```

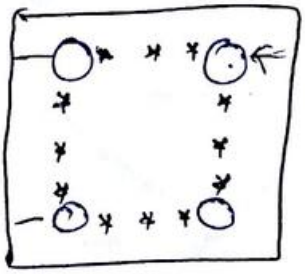
M



```
if( j==0 || j==n-1 || (i+j==n-1 || i==j) && i<=n/2 )
```

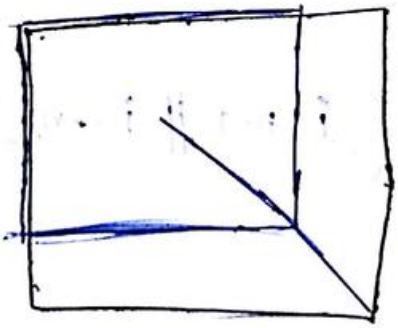


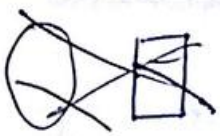
```
if( j==0 || j==n-1 || (i==j) )
```



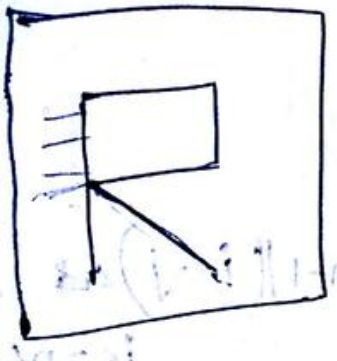
```
if( (i==0 || i==n-1) && j!=0 && j!=n/2
    || (j==0 || j==n/2) && i!=0 && i!=n-1 )
```

if



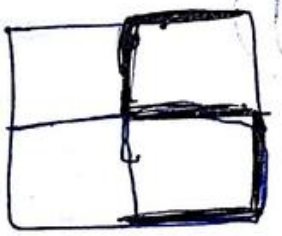


```
if( ((i==0 || i<=3*n/4) && j<=3*n/4 ||
    (j==0 || j==3*n/4) && i<=3*n/4 ||
    i==j && i>=n/2)
  Sop("* ");
  Soln();
```

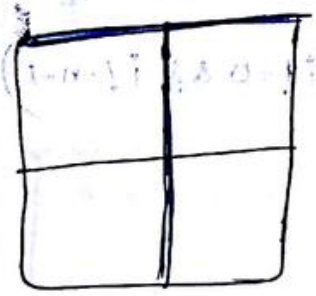


```
if ( i==0 || i==n/2 || j==0 || j==n/2 && i<=n/2
     || (i==j && i>=n/2))
```

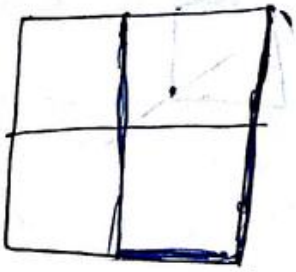
$i == n/2$ && $j <= n/2$



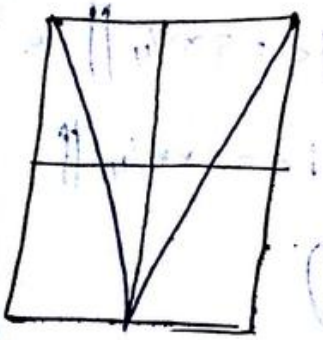
```
if (i==0 || (j==n/2 && i<=n/2) || i==n/2 || i==n-1 ||
     (j==n-1 && i>=n/2))
```



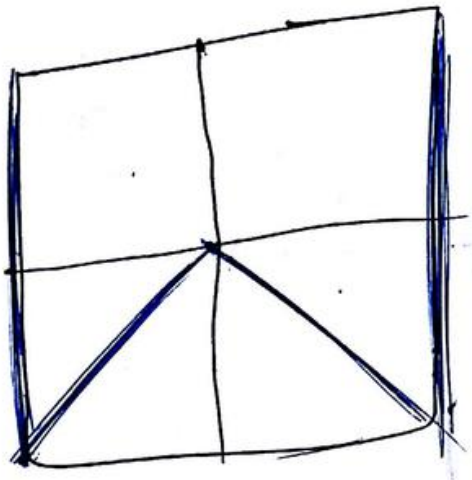
```
if (i==0 || j==n/2)
```



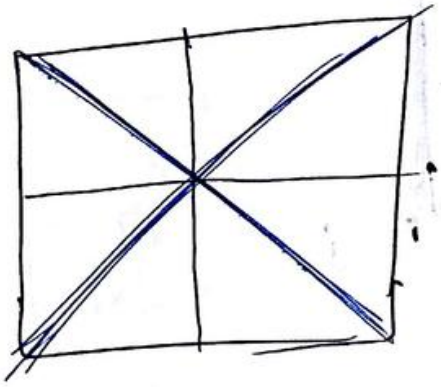
```
if ( i==n-1 || j==n/2 || j==n-1)
```



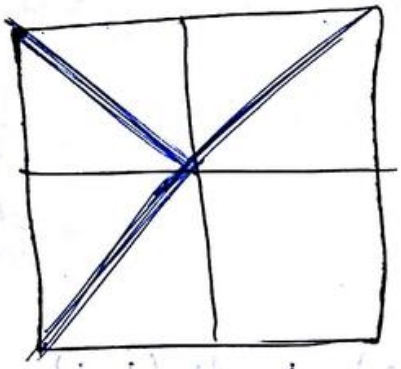
```
if ( i==j || i+j==n-1)
```



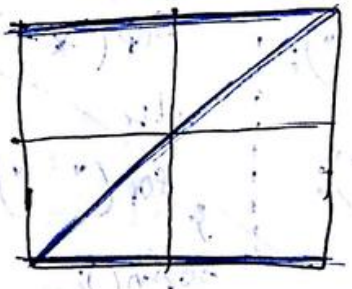
```
if( j==0 || j==n-1 ||
  ((i==j)|| i+j==n-1) && i>=n/2)
```



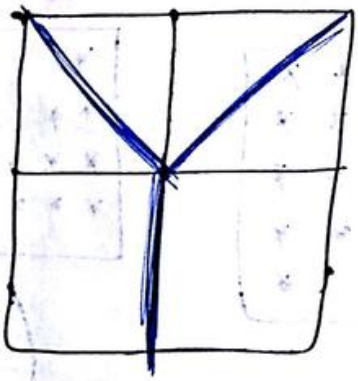
```
if( i==j || i+j==n-1)
```



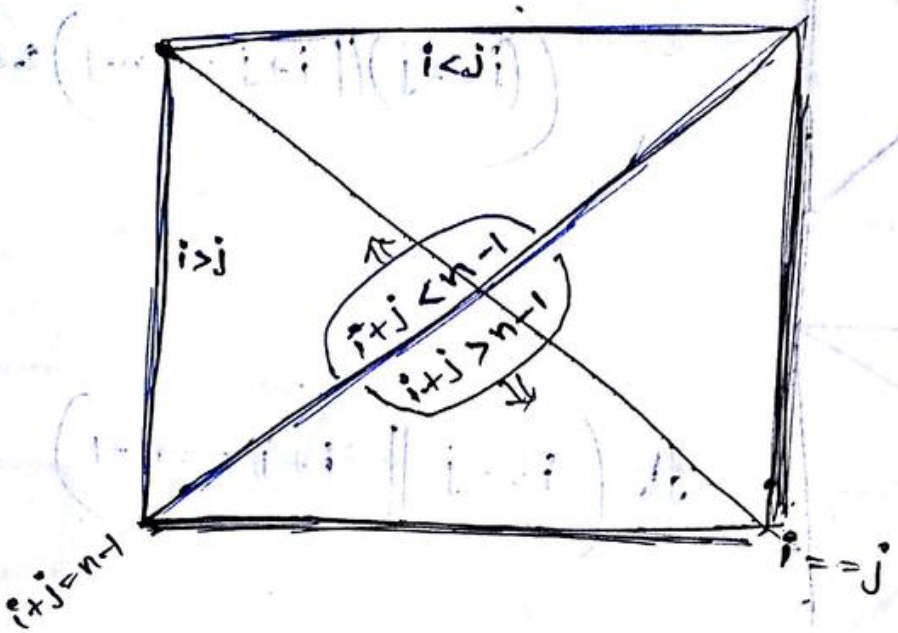
```
if(((i==j && i<=n/2) || i+j=n-1)
```



```
if( i==0 || i==n-1 || i+j==n-1)
```



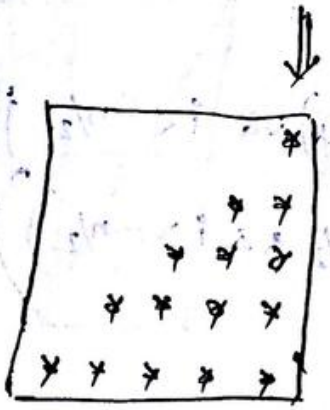
```
if( (i==j || i+j==n-1) && i<=n/2 ||
   (j==n/2 && i>=n/2))
```

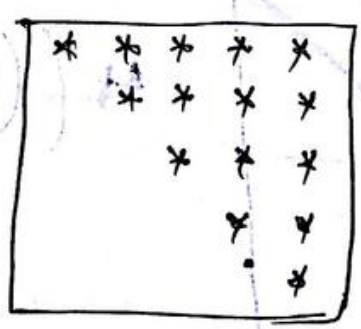


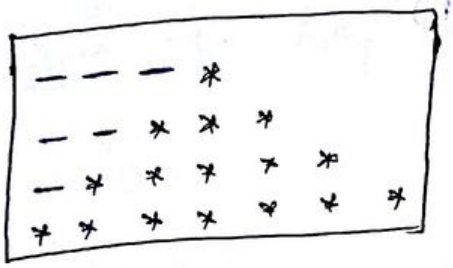

```
int n = 13;
for ( int i=0; i<n; i++ )
    for ( int j = 0 ; j<n ; j++)
    {
        if ( i+j > n-1)
        Sop ("* ");
        else
    }   Sop ("  ");
Sopln();
```

```
{ if ( i<j)
    Sop ("* ");
  else
  Sop ("  ");
}
Sopln();
```

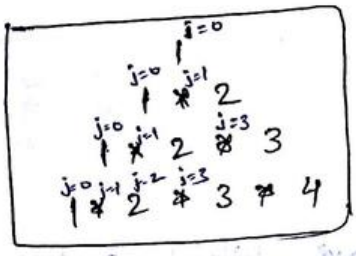
```
if (i>j)
  Sop ("* ");
  else
  Sop ("  ");
}
Sopln();
```







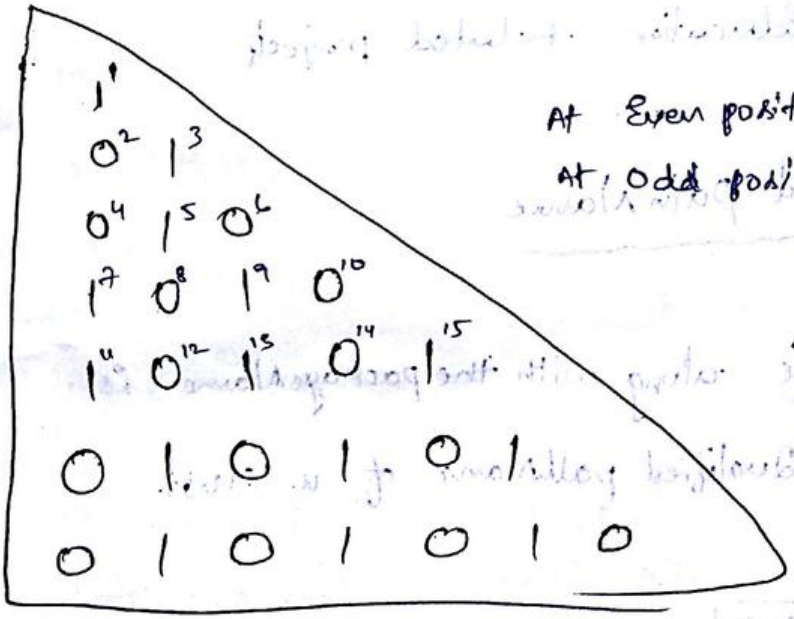
```
for(i=0; i<n; i++)
  for(j=0; j<n-1-i; j++)
    Sop(" ");
  for(j=0; j<=2*i; j++)
    Sop("* ");
```




```
for(i=0; i<=n; i++)
{
  for(j=0; j<n-1-i; j++)
    Sop(" ");
  int k=1;
  for(int j=0; j<=2*i; j++)
  {
    if(j%2!=0)
      Sop("* ");
    else
      Sop(k++);
  }
}
```

Where j value odd → print * (*)

j value Even → print start with (1)



At Even position → 0

At odd position → 1

```
int count=1;
for(i=0; i<n; i++)
{
  for(j=0; j<=i; j++)
    Sop(" " + count%2);
  count++;
}
```

WAP to check **ARMSTRONG**

→ Sum of power of ~~n~~ ~~d~~ no of digits of individual digits is called ARMSTRONG

Ex:-

① $23 \neq 2^2+3^2$

② $321 \neq 3^3+2^3+1^3$

③ $1234 \Rightarrow 1^4+2^4+3^4+4^4$

④ $12345 \Rightarrow 1^5+2^5+3^5+4^5+5^5$

⑤ $123456 \Rightarrow 1^6+2^6+3^6+4^6+5^6+6^6$

2 is Armstrong $(2^1 == 2)$

343 is Armstrong.

$(3^3+4^3+3^3) == 343$

num/10 -1

num/10 ==1

count++;

```
class ArmStrong
{
  static int IsArmStrong(int n, int count)
  { int sum=0;
    while (n>0)
    { n = n%10;
      sum = sum + pow(n, count)
      n= n/10;
    }
  } return Sum;
Public Static void main (String[] args)
{
  Sop ("Enter the num");
  Scanner sc1 = new Scanner(System.in);
  int num = sc1.nextInt();
      IsArmStrong(num) int temp = num;
    while (num>0)
    {
     //num = num%10;
      count++;
      num = num/10;
    }
    int n = IsArmStrong(temp, count);
      if (n==temp)
      Sop(temp + "is a Armstrong number");
    else
      Sop(temp + "is not a Armstrong number");
}
Static int pow (int n, int p)
    { int pro = 1;
```

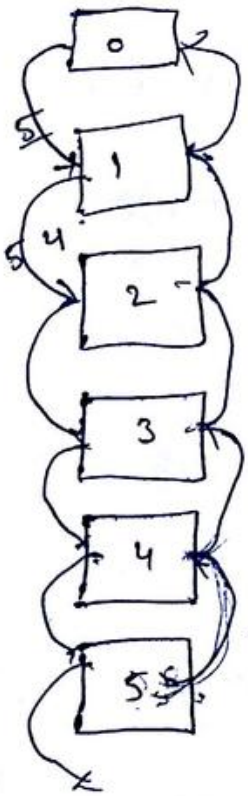
```
while (p>0)
{
    pro = pro*n;
    p--
}
return pro;
}
```

```
int n=3;
for (i=0; i<n; i++)
{
 for (j=0; j<n-1; j++)
    Sop(" ");
 for(j=0; j<=i; j++)
  if(j==0)
   Sop("*");
  else
   Sop(" ");
 for(k=0; k<i-1; k++)
  if (k==i-1)
   Sop("*");
  else
   Sop(" ");
}
 Sopln();
for(i=n-1; i>0; i--)
{
 for (j=0; j<n-i; j++)
   Sop(" ");
 for (j=0; j<=i-1; j++)
  if (j==0)
   Sop("*");
  else
   Sop(" ");
 for(k=0; k<i-1; k++)
  if(k==i-1)
   Sop("*");
} else Sop(" "); Sopln();
```

WAP to print 1 to 10 without using a loop.?

```
class Main
{
  static void value(int n)
  {
    sop(n);
  }
  psvm()
  {
    value(n-1)
  }
```

return




```
class Main
{
  static void value(int n)
  {
    if(n!=0)
    int n=value(n-1);
    sop(n)
  }
  psvm()
  {
  int n=value(10);
  sop(n);
  }
```

Recursion :- A function calling itself is called Recursion.

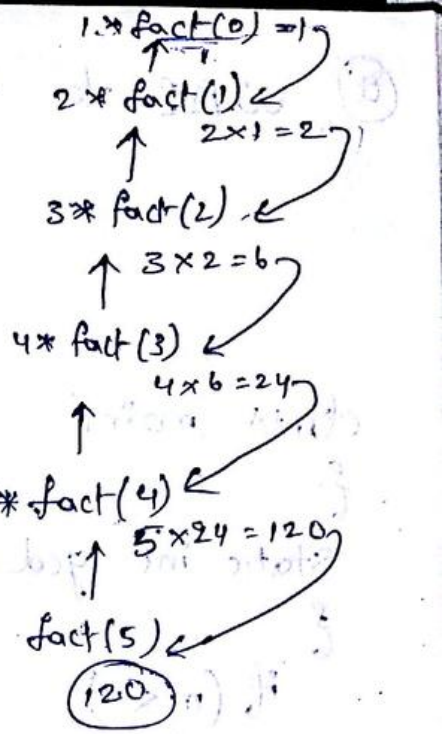
① WAP to print 1 to 10 numbers without using loop.

```
class Main
{
  static void Print(int n)
  {
    if (n>0)
    {
      Print(n-1);
    }
    Sopln(n);
  }
  Public static void main(String[] args)
  {
    sop("-----");
    print(10);
    sop("-----");
  }
```

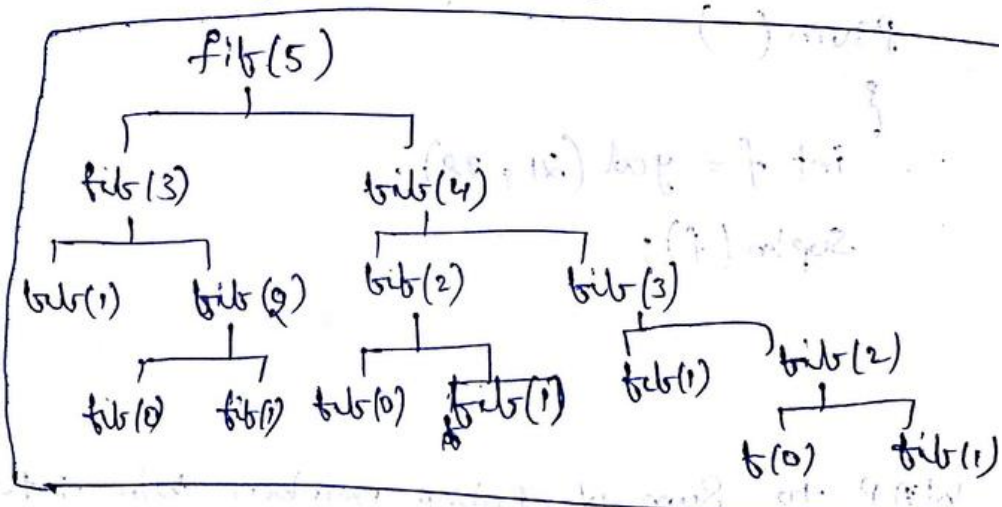
② WAP factorial of a given num Using Recursion

```
class main
{
  Static int factorial(int n)
  { int fact=1;
    while(n>0)
    {
      fact = fact * factorial(n-1);
    }
    return fact;
```

```
class main1
{
  static int fact (int n)
  {
    if (n==0)
      return 1;
    return n * fact (n-1);
  }
  psvm ( )
  {
    int f = fact (5);
    sop (f);
  }
```



WAP to fibinocci numbers using recursion.



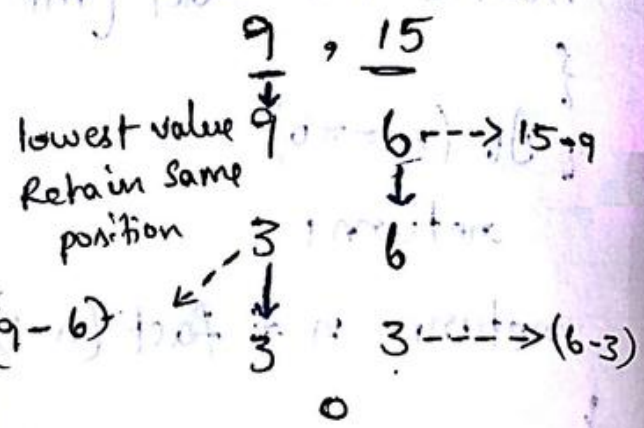

```
class main1
{
  static int fibi (int n)
  {
    if (n==0)
      return 0;
    else if (n==1)
      return 1;
    else
      return fibi(n-1) + fibi(n-2);
  }
```

Sop(fib(3) + fib(4))

fib(5)

```
psvm( )
{
  int f = fibi(8);
  Sop (f);
}
```

(a) WAP to find G.C.D of a number




```
class main1
{
 Static int gcd (int m, int n)
 {
   if (m<n)
    return gcd (n, m);
   if (n==0)
    return m;
  return gcd(n, m%n);
 }
 psum ( )
 {
   int f = gcd (21, 28)
   Sop ln (f);
```

WAP to Sum of Prime numbers b/n 1 to 100

WAP to find the Sum of multiple of 3 and 5 b/n 1 to 100

```
a
b f
c g j
d h k m
e i l n o
```

```
int i, j, k, n=5;
char ch = 'a';
for (i=0; i<n; i++)
{
  k=i;
  for(j=0; j<=i; j++)
  {
    SOP((char)(ch+k)+" ");
    k=k+n-1-j;
  }
  SOPln();
}
```

Sum of prime numbers

```
class main1
{
  static int prime(int n)
  {
    int temp=0;
    int sum=0;
    for(int i=1; i<=n; i++)
    {
      for(j=2; j<=i/2; j++)
      {
        if(i%j==0)
        {
          temp=1;
          break;
        }
      }
      if(temp==0)
      {
        sum=sum+i;
      }
    }
    return sum;
```

```
Public static void main (String [] args)
{
    Sop ("-------");
    Scanner sc1 = new Scanner (System.in);
    Sop("Enter no. of prime numbers
                you want to Sum");
    int num = sc1.nextInt();
    int sum = prime(num);
    Sop ("Sum of prime no's bln 1 and "+num+
            "is" + Sum);
}
```

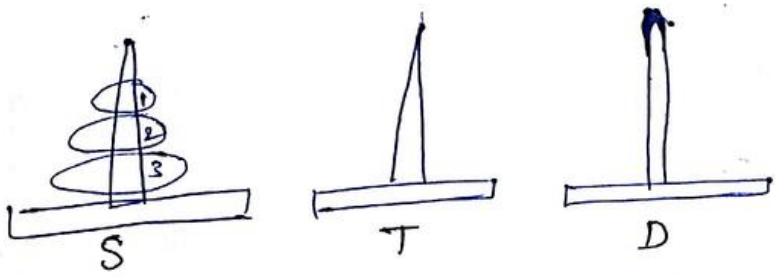
Sum of multiple 3 & 5

```
class main1
{
    static int Sum_mul (int n1 , int n2)
        { int sum=0;
          for( i=n1; i<=n2 ; i++);
          {
            if ((i%3 ==0)&&(i%5==0))
            {
              Sum = Sum + i;
            }
          }
          return Sum;
        }
    Public static void main (String [] args)
    { Sop("----"); Sop("Enter start no ");
      int num1 = sc1.nextInt();
      Sop("Enter upto which num"); int num2 = sc1.nextInt();
```

```
int Sum = Sum_mul(num1, num2);
sop("Sum of multiple of 3 & 5, in b/n "+num1+
    " and "+num2+" is "+Sum);
}
```

9/5/15

BY USING TOWER OF HENNAI



1. move S to D
2. move S to T
1. move D to T
3. move S to D
1. move T to S
2. move T to D
1. move S to D

(*) for n numbers of plates

(*) $2^n - 1$ moves

⇓

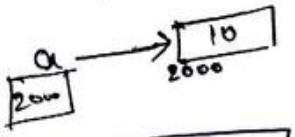
3 plates

$2^n - 1 \Rightarrow 2^3 - 1 = 7$ moves

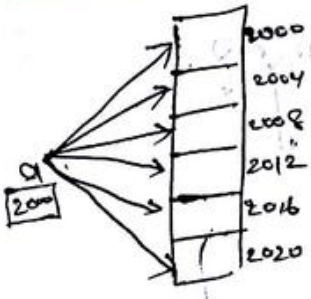
```
Void move_Disc (int n, char Src, char Temp, char des)
{
  if(n==1)
  {
    Sopln(n+"moving from"+Src+"to"+Des);
    return;
  }
  move_Disc(n-1, Src, des, Temp);
  Sopln(n+"moving from"+Src+"to"+des);
  move_disc(n-1, Temp, Src, des);
}
```

ARRAYS

`int a=10;` ⟹ here a is an identifier and it pointing to one memory location



`int a[];` ⟹ here a is an identifier and it pointing to more than one memory location and Also we make that a variable pointing to one memory location



`a[0], a[1], a[2], a[3], a[4]`

`a[5]` ⊗

`<datatype> <identifier> [];` // Array Declaration

```
int a[];
double d[];
```

`<identier> = new <datatype> [size];` // Allocation

```
a = new int[5]
d = new double[4];
```

`<identifier> [index] = values;` // Initialization

`0 => index < size`

`a[2] = 21;` ✓

`a[-1] = 35;` ⊗ at compile time

`a[5] = 45;` ⊗ at runtime

Declaration + Allocation

<datatype><identifier>[] = new <datatype>[size];

int a[] = new int[5]

double d[] = new double[4];

Declaration + Allocation + Initialization

int arr[] = {10, 20, 30, 40, 50}; // d/A/i

int mark[] = {95, 96, 97, 98, 99}; // d/A/i

How to get how many elements in an Array?

→ mark.length;

How to get Size of datatype?

Size = Sizeof(datatype);

10/5/16 Sum of Even numbers & odd numbers in a Arr

```
import java.util.Scanner;
class main1
{
  psum()
  {
    Scanner sc = new Scanner(System.in);
    Sop("Enter the number of elements");
```

```
int n = sc.nextInt();
Sop("enter "+n+" elements");
int arr[] = new int[n];
for(int i=0; i<n; i++)
{
    arr[i] = sc.nextInt();
}
int esum = 0, osum = 0, ec = 0, oc = 0;
for(int i=0; i<n; i++)
{
    Sop(i + "---->" + arr[i]);
    if(arr[i] % 2 == 0)
    {
        esum += arr[i];
        ec++;
    }
    else
    {
        osum += arr[i];
        oc++;
    }
}
Sop(ec + "Even numbers Sum" + esum);
Sop(oc + "odd numbers Sum" + osum);
}
}
```

(*) 1st Biggest & 2nd Biggest Element from a Given Array with out Sorting

```
class main1
{
  psvm()
  {
    int ar[] = {12, 34, 32, 54, 45, 6, 17};
    int n = ar.length;
    System.out.println("number of elements : "+n);
    int fbig = ar[0];
    int sbig = ar[0];
    for(int i=1; i<ar.length; i++)
    {
      if(ar[i] > fbig)
      {
        sbig = fbig;
        fbig = ar[i];
      }
      else if(ar[i] > sbig)
      {
        sbig = ar[i];
      }
    }
    Sop("1st biggest element in array is" + fbig);
    Sop("2nd biggest element in array is" + sbig);
  }
}
```

1st biggest & 2nd biggest position

```
int fbig = 0;
int sbig = 0;
for(i=1; i<n; i++)
{
  if(ar[i] > ar[fbig])
  {
    sbig = fbig;
    fbig = i;
  }
  else if(ar[i] > ar[sbig])
  {
    sbig = i;
  }
Sop("1st biggest ele pos :"+fbig);
Sop("2nd biggest ele pos :"+sbig);
```

fbig=12, sbig=12

34>12

sbig=12

fbig=34

34,12,32,54,45,6,17

32>34

sbig=32

34 32 54 45 6

54>34

sbig=34

fbig=54

54 54 45 6

45>54

sbig=45

54 45 6

(*) Reverse the Array Elements without Using Another Array!

```
class main1
{
  psvm()
  {
    int arr[] = {12,34,32,54,45,6,17}
    int n = arr.length;
    Sop("number of elements : "+n);
    int temp ;
    for(int i=0 ; i<n/2; i++)
    { temp = arrr[i]
      arrr[i] = arrr[n-i-1];
      arrr[n-i-1] = temp;
    }
    Sop("After Reversing ");
    for(i=0 ; i<n ; i++)
    {
      Sop (arrr[i]);
    }
  }
```

```
class main1
{
  Static void disp(
```

o/p :
9, 8, 7, 6, 5, 4, 3, 2, 1, 0
0, 1, 2, 3, 4, 4, 3, 2, 1, 0

```
psvm()
{
  int arr[] = new int[10];
  for(int i=0; i<arr.length; i++)
    arr[i] = 9-i;
  disp(arr);
```

```
for(i=0; i<ar.length; i++)
  arr[i] = arr[ar.length-1-i];
Sop(
disp(arr);
```

Assignment

(*) Sum & Avg of Array Elements

(*) 1st least & 2nd least element from the Array without sorting

(*) WAP to find the sum of even positions & odd position elements.

(*) Sum & Avg of Array

```
import java.util.Scanner;
class main
{
  psvm( )
  {
    Scanner sc = new Scanner (System.in);
    Sop ("Enter the no of Elements");
    int n = sc.nextInt();
    Sop ("Enter "+n+" Elements");
    int arr[] = new int[n];
    int sum=0;
    for (i=0 ; i < arr.length ; i++)
    {
      arr[i] = sc.nextInt;
      sum = sum + arr[i];
    }
    Sop ("Sum of "+n+" Elements" +Sum);
    Sop ("Avg of "+n+" Elements" +(Sum/arr.length));
  }
}
```

✱ 1st least & 2nd least Element from the Array without sorting

```
import java.util.Scanner;
class main1
{
 psvm()
 {
  Scanner sc = new Scanner(System.in);
  Sop("Enter the no. of Elements");
  int n = sc.nextInt();
  Sop("Enter "+n+" Elements");
  int ar[] = new int[n];
  int fl = ar[0], sl = ar[0]
  for(i=1; i<n; i++)
  {
   if(ar[i] < fl)
   {
    sl = fl;
    fl = ar[i];
   elsif(ar[i] < sl)
    sl = ar[i];
  }
  Sop("1st least elem from Array" + fl);
  Sop("2nd least Element from Array" + sl);
 }
}
```

⊛ Sum of Even & odd position Elements in an Array

```
class main1
{
  psum ( )
  { int ar[] = new int[n];
    Scanner sc = new Scanner (System.in);
    Sop("Enter no of Elements");
    int n = sc.nextInt ();
    Sop("Enter the Elements of the Array");
    for ( i=0; i<n ; i++)
    {
      ar[i] = sc.nextInt();
    }
    int es=0, os=0;
    for ( i=0; i<n; i++)
    {
      if ( i%2 == =0)
        es = es + ar [i];
      else
        os = os + ar [i];
    }
    Sop("Sum of Even position Elements" + es);
    Sop("Sum of odd position Elements" +os);
  }
}
```

11/5/16

* SWAP the 1st part Array Element with the 2nd part

Ex:-

ar[] = {2,3,4,5,6,7,8}

o/p ⇒ 6 7 8 5 2 3 4

```
class main1
{
  psvm( )
  {
    int ar[] = {2,3,4,5,6,7,8};
    int n = ar.length;
    int temp;
    for(i=0; i<n/2; i++)
    {
      temp = a[i]
      a[i] = a[n-n/2+i]
      a[n-n/2+i] = temp;
    }
    SOP("After Swapping");
    for(i=0; i<n; i++)
    SOP(a[i]);
}
```

0 1 2 3 4 5 6

7-1-3
7-0-3
4

7-1-3
7-4
3

7-(i+1)-3
7-1

7-3
(3+1)
-1
4+1

a[0] = a[7-3+0]
a[0] = a[4]

a[1] = a[7-3+1]
a[1] = a[5]

(*) INSERT the Element into Existing Away in Specified position

```
class main1
{
  static int[] insert(int a[], int ele, int inx)
  {
    if(inx<0 || inx>=a.length)
    {
      Sop("index out of bound");
    }return a;
    int na[] = new int[a.length+1];
    for(i=0; i<inx; i++)
      na[i] = a[i];
    na[inx] = ele;
        for(i=inx; i<a.length; i++)
          na[i+1] = a[i];
      }return na;
psvm()
{
  Sop("Enter no. of Elements");
  int n = sc.nextInt();
  int ar[] = new int[n];
  Sop("Enter Elements into Array");
  for(int i=0; i<ar.length; i++)
    ar[i] = sc.nextInt();
  ar = insert(ar, 9, 3);
  for(i=0; i<=n; i++)
    Sop(ar[i]);
}
```

(*) DELETE the Elements from Existing Array in Specified position

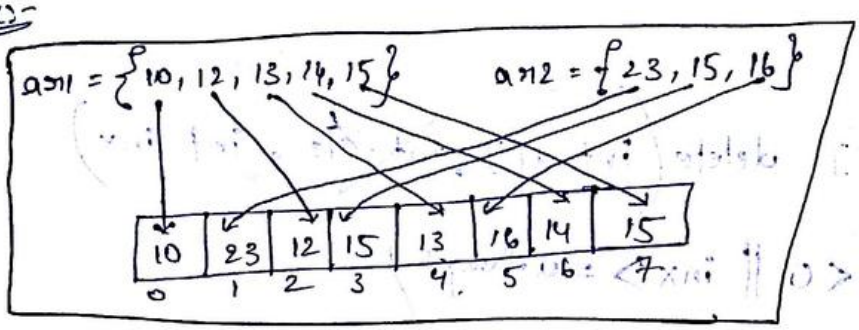
```
class main1
{
  static int[] delete(int a[], int inx)
  {
    if(inx<0 || inx>=a.length)
    {
      Sop("index out of bond");
      return a;
    }
    int na[] = new int[a.length-1];
    for(i=0; i<inx; i++)
      na[i] = a[i];
    for(i=inx+1; i<a.length; i++)
      na[i-1] = a[i];
    return na;
  }
  public static void main(String[] args)
  {
    Sop("Enter number of Element you want");
    int n = sc.nextInt();
    int ar[] = new int[n];
    Sop("Enter Elements into Array");
    for(int i=0; i<ar.length; i++)
      ar[i] = sc.nextInt();
    ar = delete(ar, 2);
    for(i=0; i<n-1; i++)
      Sop(ar[i]);
  }
```

I/P
Enter number of ele you want
5
Enter elements into Array
1, 2, 3, 4, 5
⇒ 1 2 4 5 → After delete one index

⇒ After delete one index from Array

(*) MERGE 2 Arrays in Zig Zag manner

Ex:-




```
class main1
{ psvm()
 {int arr1[] = { 12, 34, 5, 67, 13 };
  int arr2[] = { 10, 7, 61, 2 };
  display(arr1);
  SOP("----------");
  display(arr2);
  int na[] = new int[arr1.length + arr2.length];
  int i, K;
  SOP("----------");
  for(i=0,K=0; i<arr1.length && i<arr2.length; i++)
  {
    na[K] = arr1[i]
    K++;
  ; na[K] = arr2[i]
  for( ; i<arr1.length ; i++,K++)
      na[K] = arr1[i];
  for( ; i<arr2.length ; i++,K++)
     na[K] = arr2[i];
  display(na);
 }
 static void display(int a[])
 { for(int i=0; i<a.length ; i++)
     SOP(a[i]+",");
 } Soln();
```

(*) By Comparing 2 Arrays - find Distinct and Common Elements in the Array

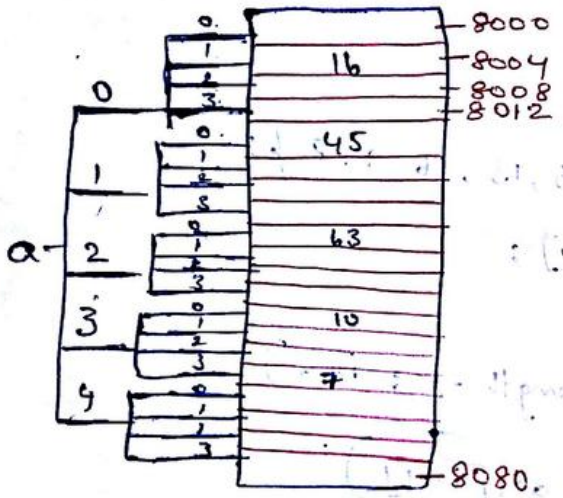
```
class main1
{ static void display(int a[])


public static void main(String[] args)
{ int arr1[] = {12,34,5,67,13};
  int arr2[] = {34,56,78,12};
  display(arr1);
  Sop("----------");
  display(arr2);
  Sop("Common Elements :");
  for(int i=0; i<arr1.length; i++)
  {for(int j=0; j<arr2.length; j++)
    { if(arr1[i]==arr2[j])
      { Sop(arr1[i]+" ");
        break;
      }
    }
  }
Sopln();
Sopln("Distinct Element");
for(int i=0; i<arr1.length; i++)
 { int find=0;
   for(int j=0; j<arr2.length; j++)
     {
```

```
            if(arr1[i] == arr2[j])
            {
                find = 1;
                break;
            }
        }
        if(find == 0)
            Sop(arr1[i] + " ");
    }
    for(int i=0; i<= arr2.length; i++)
    {
        int find = 0;
        for(int j=0; j< arr1.length; i++)
        {
            if(arr2[i] == arr1[j])
            {
                find = 1;
                break;
            }
        }
        if(find == 0)
            Sop(arr2[i] + " ");
    }
}
```

(*) Highest Contigious Sum of 2 Element from the Array

```
class main1
{
  psum()
  {
    int arr[] = {10, 3, 2, 15, 13, 16, 9, 8};
    int Sum = arr[0] + arr[1];
    int k=0;
    for(int i=1; i<arr.length-1; i++)
    {
      if(Sum < arr[i] + arr[i+1])
      {
        Sum = arr[i] + arr[i+1];
        k=i;
      }
    }
    SOP(Sum);
    SOP(k + "--->" + arr[k]);
    SOP(k+1 + "--->" + arr[k+1]);
  }
}
```

2-D Array [2 Dimensional Array]




```
int a[][];
arr = new int[5][4];
```

No of Row — No of Element

- ⇒ a[0][1] = 16
- ⇒ a[1][0] = 45
- ⇒ a[2][0] = 63
- ⇒ a[3][0] = 10
- ⇒ a[3][3] = 7

(*) arr.length ⟹ No of 1-D Rows

(*) ↳ arr[0].length ⇒
↳ arr[1].length ⇒ } No. of elements in
↳ arr[2].length ⇒ that Row.

Ex:-

```
int a[][] = {{1,2,3}, {4,5,6}, {7,8,9,10}, {1,8}};
              0        1        2           3
```

→ arr.length 4

→ arr[0].length 3

→ arr[1].length 3

→ arr[2].length 4

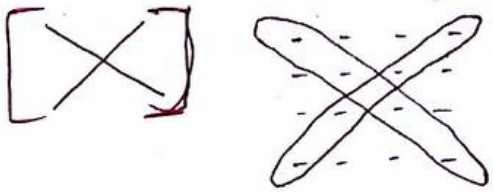
→ arr[3].length 2

```
class main1
{
  psvm( )
  {
    int ar[][] ={{17,18,20,22},{31,32,45},{8,6,7,4,3}};

    for(int i=0; i<ar.length; i++)
    {
      for(int j=0; j<ar[i].length; j++)
      {
        Sop(" "+ar[i][i]+"("+i+","+j+") ");
      }
      Sopln();
    }
  }
}
```

(*) Biggest Element in the matrix Row wise & Column wise

```
class main1
{
 static void display(int mat[][])
  {
   for(int i=0; i<mat.length; i++)
    {
     for (int j=0; j<mat[i].length; j++)
       Sop(mat[i][j] + " ");
    Sopln();
  } }
Psum()
{ int arr[][] ={{17,38,20,22}, {31,32,45,17},
                {18,66,27,14}, {12,34,45,32}};
  display(arr);
  for(int i=0; i<arr.length; i++)
   {
    int rbig = arr[i][0];
    int cbig = arr[0][i];
    for(int j=1; j<arr[i].length; j++)
     {
       if(rbig < arr[i][j])
        rbig = arr[i][j];
       if(cbig < arr[j][i])
     }    cbig = arr[j][i]
  Sop(i+1+" row bigest element "+rbig);
  Sop(i+1+"colum bigest element"+cbig);
```

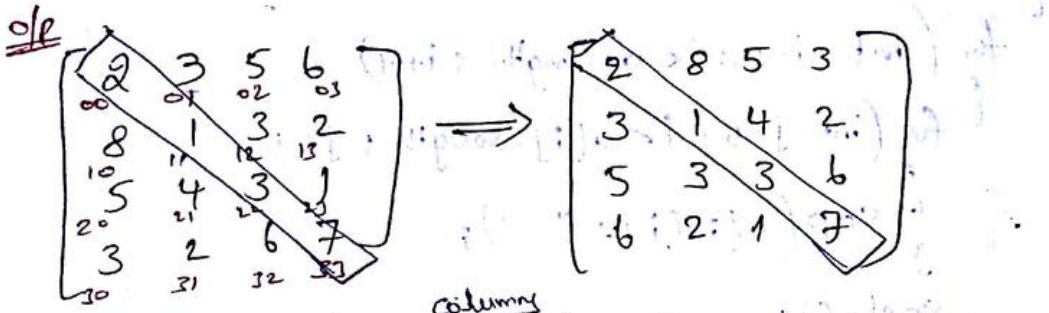
(*) Biggest Element in the matrix

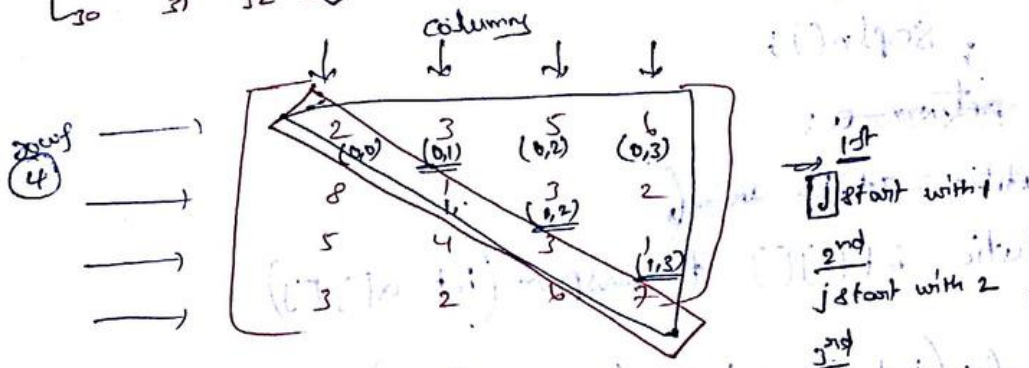
Diagnals 

```
class main1
{

psum()
{int ar[][] ={{

display(ar);
int pbig = ar[0][0];
int sbig = ar[0][ar[0].length-1];
for(int i=0; i< ar.length ; i++)
{ for(int j=1 ; j<ar[i].length ;j++)
  { if(i==j)
      if(ar[i][j] > Pbig)
      Pbig = ar[i][j];
  if(i+j == ar.length-1)
      if(ar[i][j] > Sbig)
  } } Sbig = ar[i][j];
```

(*) TRANSPOSE the Given matrix.

o/p






```
import java.util.Scanner;
class main
{
  psum (----)
  {
    Scanner sc = new Scanner(System.in);
      Sop("Enter the order of Square matrix");
      int n = sc.nextInt();
      int mat[][] = new int[n][n];
     Sop("Enter "+n*n+" Elements");
    for(int i=0; i<n ; i++);
    {
      for(int j=0; j<n ; j++)
        {
          mat[i][j] = sc.nextInt();
        }
    }
  Sop("Given matrix");
  display(mat);
  mat = transpose(mat);
  Sop("After transpose"); display(mat);
```

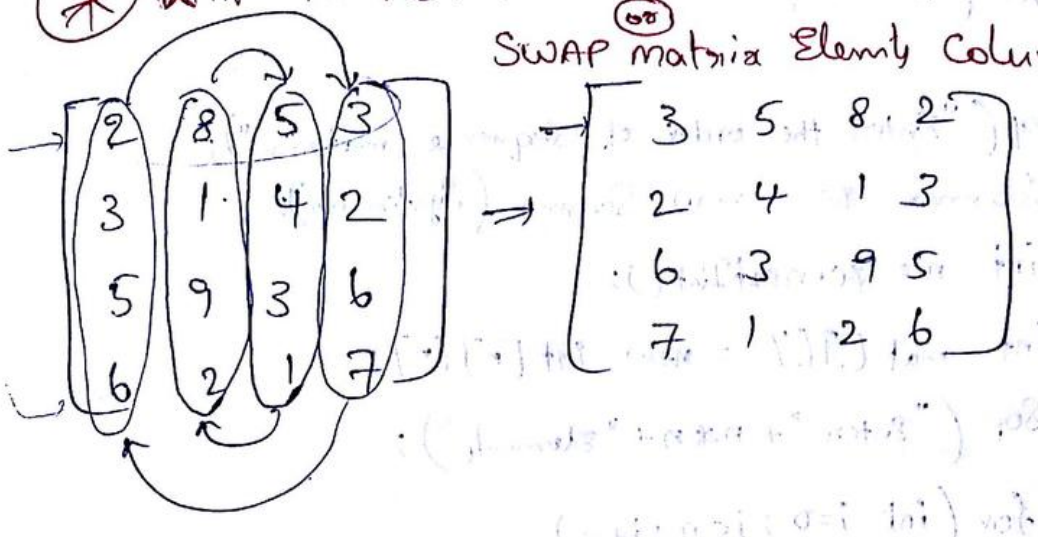
```
Static void display(:int _a[][])
{
  for(int i=0.; i<a.length ; i++)
  {
    for(int j=0; j<a[i].length ; j++)
    {
      SOP(a[i][j]+" ");
    }
    Sopln();
  }
}
```

~~return a;~~

~~public static void~~

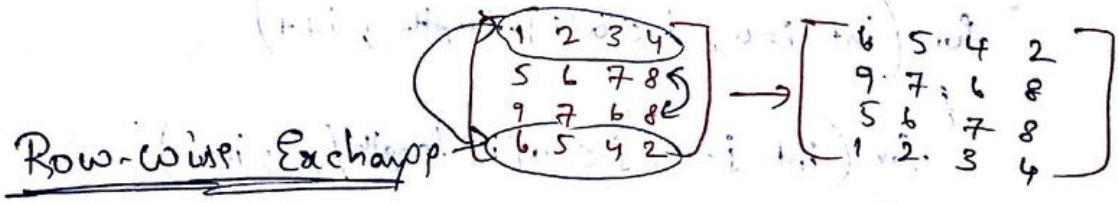
```
static int[][] transpose(int a[][])
{
  for(int i=0; i<a.length ; i++)
  {
    for(int j=i+1 ; j<a[i].length ; j++)
    {
      int temp = a[i][j]
      a[i][j] = a[j][i]
      a[j][i] = temp;
    }
  }
  return a;
}
```

(*) WAP to Reverse matrix Elements row wise (or) SWAP matrix Elemts Colum wise




```
class main1
{
   static void display (int a[][])
   {
     for (int i=0 ; i< a[].length ; i++)
      {
        for(int j=0 ; j < a[i].leyth ; j++)
            sop(a[i][j] + " ");
      } Soplnc();
   }
   static int[][] rev (int a[][])
   {
     for (int i=0; i< a[i].length ; i++)
      {
        for (int j=0; j< a[i].length/2 ; i++)
          {
            ft(a[i][j] != a[i][a[i].length-1-j])
            {
              int temp = a[i][j]
                a[i][j] = a[i][a[i].length-1-j];
              a[i][a[i].length-1-j] = temp;
            }
          }
      }
     return a;
```

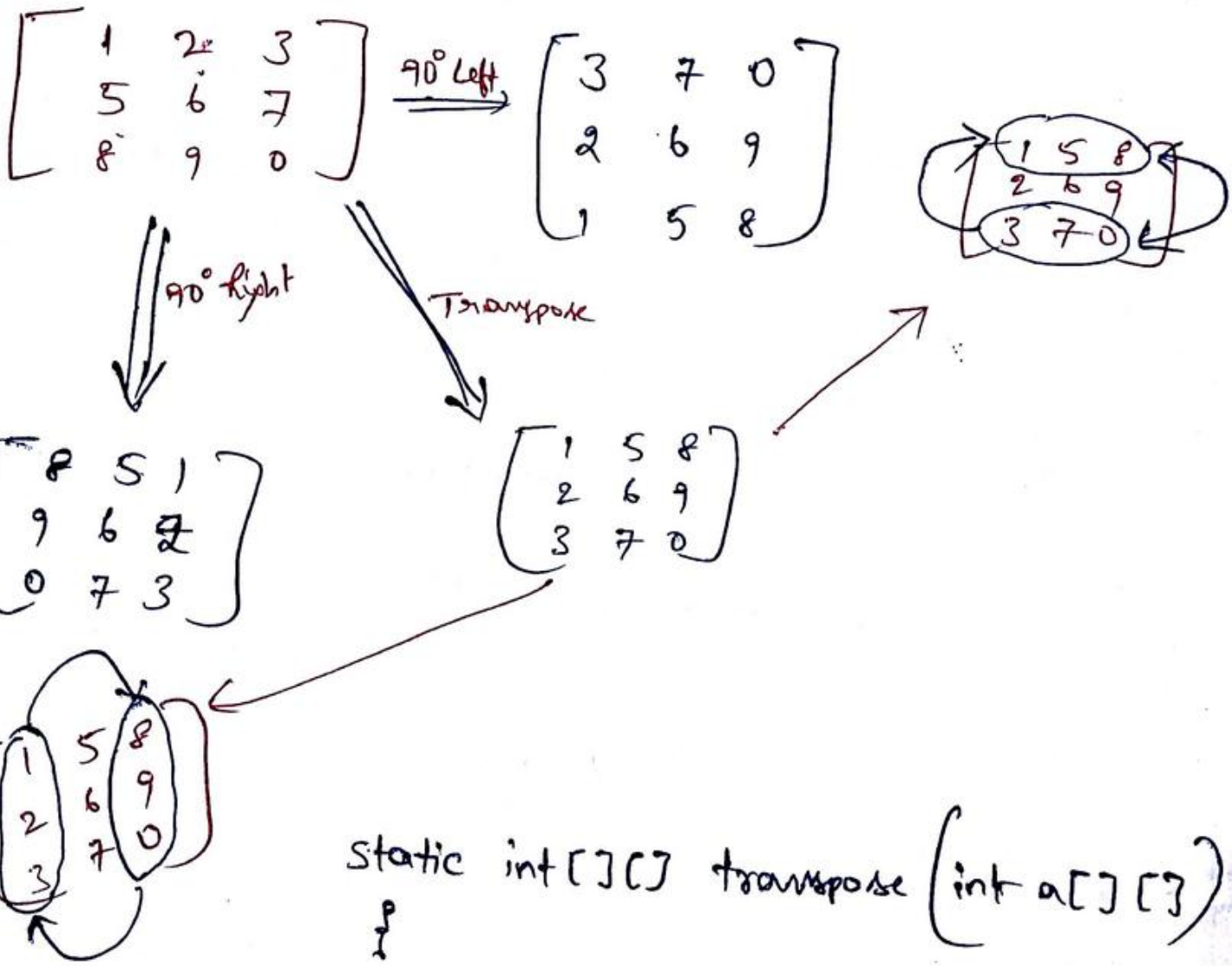
```
psvm (----)
{
    Sop("Enter the order of Sequence matrix");
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
    int mat[][] = new int[n][n];
    Sop("Enter "+ n*n +" elements");
    for(int i=0; i<n; i++)
    {
      for(int j=0; j<n; j++)
      {
        mat[i][j] = sc.nextInt();
      }
    }
    Sop("Given matrix");
    display(mat)
    mat = rev(mat)
    Sop("After rev matrix");
    display(mat);
}
```



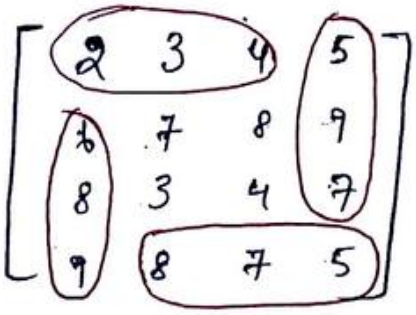
Row-wise Exchange

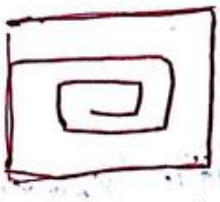
```
Static int[][] rev(int i a[][])
{
    for(int i=0; i<a.length/2; i++)
    {
      for(j=0; j<a[i].length; j++)
      {
        int temp[0] = a[i][j]
        a[i][j] = a[a.length-1-i][j]
        a[a.length-1-i][j] = temp[0];
      }
```

(*) Rotate the Given matrix into 90° Right & 90° Left



17/5/16 (*) Display the matrix in Spiral order





```
class main
{
  psvm (---)
  {
    int mat[][] = {{2,3,4,5},{6,7,8,9},{8,3,4,7},{9,8,7,5}};

    for(int i=0, j=mat.length-1; i<j; i++, j--)
    {
      for(int k=i; k<j; k++) Sop(mat[i][j] + " ");
      for(int k=i; k<j; k++) Sop(mat[i][j] + " ");
      for(int k=j; k>i; k--) Sop(mat[j][k] + " ");
      for(int k=j; k>i; k--) Sop(mat[j][k] + " ");
    }
    if(n%2 != 0)
    Sopln(mat[mat.length/2][mat.length/2])
  }
}
```

(*) Convert the Upper Case letter into lower Case

Lower Case to Upper Case

```
import java.util.Scanner
class Main
{
  psvm (----)
  {
    Scanner sc1 = new Scanner(System.in);
    Sop("Enter your String");
    String st = sc1.nextLine();
    Sop(st);
    char ch[] = st.toCharArray();
    for(int i=0; i<ch.length; i++)
    {
      if(ch[i]>=65 && ch[i]<=90)
        ch[i]=(char)(ch[i]+32);
      else
        if(ch[i]>=97 && ch[i]<=122)
          ch[i]=(char)(ch[i]-32);
    }
    st = new String(ch);
    Sop(st);
  }
}
```

nextLine ⇒ taking String value with Spaces Also

javap java.lang.object ⟹ It gives Skeleton of object class

javap java.lang.String ⟹ It gives Entire methods & Constructors which is present in String library

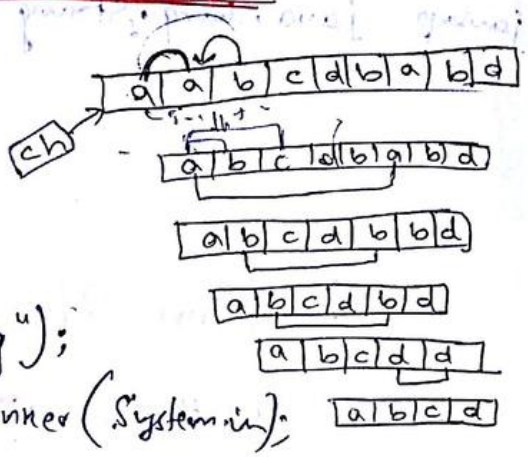
(*) Given String is PALINDROME or not with out Reversing a String

Ex)

S	A	V	A	S

```
class main
{
 psvm (- --)
 {
 Sop("Enter your String");
 String st = sc.nextLine();
 Sop(st);
 char ch[] = st.toCharArray();
 int f=1;
 for(int i=0 ; i< ch.length/2 ; i++)
  {
    if(ch[i] != ch[ch.length-1-i])
      {
        f=0;
        break;
 } }
 if(f==1)
 Soprintf("%s is palindrome" + st);
else
 Soprintf("%s is "palindrome" + st);
```

18/5/16 (*) Count the num of occurances of each charecter in a given String



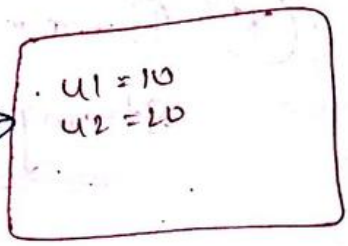

```
class main1
{
  psum (- - - -);
  {
  Sop ("Enter your String");
  Scanner sc = New Scanner (System.in);
  String st = sc.nextLine();
  char ch[] = st.toCharArray();
  int n = ch.length;
 for(int i=0 ; i<n ; i++)
  {
     int count = 1;
      for (int j= i+1 ; j< n ; j++)
       {
          if (ch[i] == ch[j])
           {
              for(int k=j ; k < n-1 ; k++)
               {
                  ch[k] = ch[k+1];
               }
              n--;
              j--;
           }
       }
   Sop( ch[i] + " --> " + count);
}
```

abababcdefgabc

a → 3
b → 3
c → 2
d → 1
e → 1
f → 1
g → 1

(*) Remove the Repeated charecter from String Except Space irrespective of cases

```
class main(
{
  psvm(---)
  {

    String st = sc.nextLine();
    char ch[] = st.toCharArray();
    int n = ch.length;
    st = "";
    for(int i=0; i<n; i++)
    {
      st = st + ch[i];
      for(int j=i+1; j<n; j++)
      {
        if(ch[i]!=' ' && (ch[i]==ch[j] ||
                ch[i]==ch[j]+32 || ch[i]==ch[j]-32))
        {
          for(int k=j; k<n-1; k++)
          {
            ch[k] = ch[k+1];
          }
          n--;
          j--;
        }
      }
    }
    sop(st);
  }
```



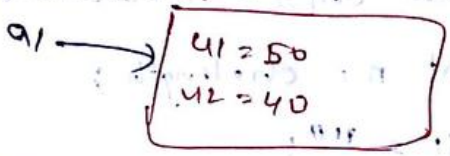

```
class A
{
    final int u1;
    final int u2;
    A (int u1 , int u2)
    {
     this.u1 = u1;
     this.u2 = u2;
    }
}
```

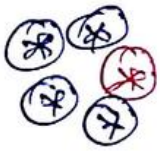
to make immutable

```
A a1 = new A (10, 20);
a1.u1 = 45;
a1.u2 = 200;
```

(X)

```
a1 = new A (50, 40);
```





count
No. of words in a given String.

32 is Ascii value of Space

```
class main1
{
 psvm (---){
  {
   SOP ("Enter your String");
   Scanner sc = new Scanner (System.in);
   String st = sc.nextLine ();
   char ch[] = st.toCharArray ();
   int n = ch.length;
   int count = 1;
   for(int i=0; i<n-1; i++
   {
    if (ch[i]==' ' && ch[i+1] !=' ' && ch[i+1] !=',' && ch[i+1]
                                                    !='.')
 || ch[i] ==',' && ch[i+1] !=' ' && ch[i+1] !=',' && ch[i+1] !='.'
```

```
|| ch[i]=='.' && ch[i+1]!='.' && ch[i+1]!=' ' && ch[i+1]!=',')
    count++;
  }
 }
}
```

(*) Count of no. of vowels, Consonants, numbers and Special charecters.

```
for(int i=0; i<n; i++)
{ if(ch[i]=='a'||ch[i]=='A'||ch[i]=='e'||ch[i]=='E'||ch[i]=='i'||
     ch[i]=='I'||ch[i]=='o'||ch[i]=='O'||ch[i]=='u'||ch[i]=='U')
   count1++;
  else if((ch[i]>='a' && ch[i]<='z')||(ch[i]>='A' && ch[i]<='Z'))
   count2++;
  else if(ch[i]>='0' && ch[i]<='9')
   count3++;
  else
   count4++;
}
Sop("Vowels --->" + count1);   Sop("Numbers --->" + count3);
Sop("Consonants --->" + count2); Sop("Specialcharacters--->" + count4);
```

19/5/12

Inner Class

class containing

```
class A
{
 int a=10
Static int b=20;
void m1()
 {
 }
Static void fy1()
 {
 }
Static
 {
 }
 {
 }
class B
{
}
Static class B
{
}
}
```

Datamembers

methods

Blocks

subclasses

Types
1. STATIC inner class
2. Non-Static inner class
3. Local inner class
4. Anonymous inner class

(*) What are the different types of membery we have in class ?

1. Static
   - Variables
   - methods
   - Blocks
   - subclass (inner class)
2. NonStatic
   - Variables
   - methods
   - Blocks
   - subclasses (inner class)

Inner class :-

Whenever class prefixed by static then we can call that class is Inner class (or) Nested class.

(*) A class inside Another class.

STATIC inner class

```
class Sample
{
  int a=30;
static int b=40;
static class Demo
   { int a1=40;
static int b1=56;
    void m()
     {
       ==
     }
   }
}
```

```
SOP(Sample.b)
Sopln(Sample.Demo1.b1)
Sopln(new Sample().a)
Sample.Demo sd =new Sample.Demo();
Sop(sd.a1);
```

NOTE:-

* ~~Note~~ Through inner class we can Access private members of outer class members

If we want to Access only Data members of inner class, No need to create instance of outer class

Static Inner Class

(1) If you define a cls inside another class prefixed by a static keyword is called static inner class
(or) Nested class

(2) Static inner class can possible to have (i) Static members
(ii) Non-Static members

(3) Static members of static inner class can possible to Access without Creating object of outer class as well as inner class

④ Inside static inner class, we can Access outer class static data members including private but not possible to Access instance members of outer class

⑤ Static members of nested class we can possible to Access. outerclass name . innerclass name . static member

⑥ If you want to Access instance member of static inner class no need to create a object for outer class

---

```
public class Sample
{
  int a=10;
  static int b=20;
  static class Demo
      {
        int a1=40;
        static int b1=50;
        void m1()
        {
          SOP(b);
          SOP(new Sample().a);
        }
        SOP("i am m1 of demo");
      static void m2()
        {
          SOP("i am m2 of demo");
        }
      }
    void msg()
    {
      SOP(Demo.b); SOP(new Demo().a1);
    }
```

(1) Inside a Instance inner class, we are not possible to Define the static members, only Non-Static members

(2) We can Access static mem's of outer class with out creating Instance of outer class

(3) We can Acesses Static & Non-static members of outer class directly with out creating instance of outer class

(4) If you want to Access instance members of Instance inner class in the outer class, we have to create instance of inner class

(5) If you want to Access instance ~~inner class~~ mem's of instance inner class, we have to Create instance of inner class as well as outerclass.

Non-Static Inner class

```
Public class Sample
{
 int a=10;
 static int b=20;
 class Demo
 {
  int a1=40;
  static int b1=50;
  Void m1()
  {
   SOP(b);
   SOP(a);
  }SOP("i'm m m1 of Demo");
  Static Void m2()
  {
   SOP("i'm m2 of Demo");
  }
  Void msg()
  {
   SOP(new Demo().a1);
   new Demo().m1();
  }
 }
class main
{
 PSVM(---)
 {
  SOP("----");
  Sample s1 = new Sample(); // outer class object
  Sample.Demo d1 = s1.new Demo(); // inner class object
  SOP(d1.a1); d1.m1();
}}
```

Instance inner class

20/5/16

LOCAL INNER CLASS

We can Create in
- methods
- Blocks

```
class Sample
{
  void m1()
  {  private int a=20;  ⊗
     static class A
     {  int a;  static int b;  ⊗
        void m2();
        {  ≡
        }
     }
     A a1 = new A();
     Sop(a1.a);
  }
```

In local class, we can't create

`A a1 = new A();` ⊗ we can't create instance of local class in out side method

1. Define a cls, Inside a method or blocks that type of class is called
2. If you want to invoke method of local inner class
   (or) If you want to Access the member of local inner class we have to instantiate the local inner class inside that method only
3. Local inner class can't be invoked from out side the method.
4. Local Inner class members must be a Non static
5. Inside a local inner class, we can Access outer class instance members
6. for local variables of local inner class, we can't use Access Specifiers

(7) Non-final local variables are Not possible to Access inside the local inner-class

Anonymous local inner class

```
Public class Main
{
  Psublic static void main (String args[])
  {
    SOP ("---------");
  Student st1 = new Student (123, "Rakesh", 73.28)
  SOP (st1);
  SOP (st1.hashCode());
  Student st2 = new Student (124, "suresh", 53.83)
  {
    public String toString()
    {
      return "hi suresh";
    }
    public int hashCode()
    {
      return id;
    }
    void m1()
    {
      SOP ("I am m1 of Anonymous");
    }
  }.m1();
}
```

object

```
object.m1();
```

```
package com.jspiders.java

public class Student
{
  int id
  String name;
  double per;
  public Student(int id, String name, double per)
  {
    super();
    this.id = id;
    this.name = name;
    this.per = per;
  }
  public String toString()
  {
    return "Student [" + name + "]";
  }
}
```

Blocks

types

① SIB (Static Initialiser Block)
② IIB (Instance Initialiser Block)

static { == }

{ == }

Def:

IIB — Instance Initialiser Block.

```
class A
{
  {
   Sopln("I am IIB of class A");
  }
  A(int a)
  { super(); ⟶ implicitly
  }
  A(int a, int b)
  { super(); ⟶ implicitly
  }
  A()
  { super(); ⟶ implicitly
  }
}
class Main1
{
  Psvm(---)
  {
    A a1 = new A();
    A a2 = new A(10);
    A a3 = new A(10,20);
  }
}
```

(*) After the compilation, IIB gets copied Aftr the Execution Super calling stmt in Every Constructor

(*) If you Execute Any task immediately After Creating an instance of that class, use IIB in a r class

o/p

I am IIB of class A
I am IIB of class A
I am IIB of class A

① IIB is used for initialize the default values of non-static data members of that class.

(*) Here we get o/p

0
0

defaultly IIB gives default values to the Data mem's of class.

```
class A
{
  int a;
  int b;
  Sop(a);   0
  Sop(b);   0
}
class main1
{ psvm(-)
  { A a1 = new A();
  }
}
```

② IIB is used for initialize the Instance final variable

(*) We can Initialize the Instance final variable Either

1. non-static-Block (IIB)
2. Constructor
3. Declaration time: ---> `final int K = 10;`

①

```
public class psample
{
  {
   Sop("welcome to Psample");
  }
  Psample()
  {
   Sop("i am Psample cons...");
  }
}
```

③

```
public class Main
{
  { Sop("I m iib of main");}
Psvm(-----)
{ Sop("mms");
  Sample s1 = new Sample();
```

②

```
class Sample Extends Psample
{ int a,b;
  final int f;
  double pi;
  {
    f = 900;
    a = 20;
    b = 90;
    pi = 3.143;
  Sop("welcome to Sample");
  }
  Sample()
  { Sop("i am no arg cons of. Samp");
  Sample(int a)
  { Sop("i am 1 arg cons of Samp");}
  Sample(int a, int b)
  { Sop("i am 2 arg cons of Samp");
```

```
Sop("--------");
Sample s2 = new Sample(10);
Sop("-------");
Sample s3 = new Sample(10,20);
}
}
```

op

mms

i am PSample cons --

(1)

```
class parent
{
  static
  { Sop("i am sib of parent");
  }
  {
   sop("i am iib of parent");
  }
  parent()
   { Sop("i am cons of parent");
   }
}
```

(2)

```
class child Extends parent
{
  static
  { Sop("i am sib of child");
  }
  {Sop("i am iib of child");
  }
child()
{
 Sop("i am cons of child");
}
child (int a)
{ this();
 Sop("i am 1 arg cons of child");
}
child (int a, int b)
{
 Sop("i am 2 arg cons of child");
}
}
```

(3)

```
class main2
{
  {
   Sop("i am iib of main2")
  }
 static
 {
  Sop("I am sib of main2");
 }
 psvm(------)
 {
  Sop("mms");
  child c1 = new child(10,20);
  Sop("-------");
  child c2 = new child(10);
  Sop("--------");
  child c3 = new child();
  Sop("--------");
```

main2 m2 = new main2();

o/p

I am sib of main 2

mms

i am sib of parent

i am sib of child

i am iib of parent

STATIC BLOCKS SIB

(1) SIB is used for initialize the default values of static data members of that class

(*) We can initialize static final variable either

(1) Declaration time

(2) Static-Blocks (SIB)

(*) If you Execute Any task only one time when creating an instance of Specific class. through out Application, use SIB in your class

(1) TO find the middle Element from the given String..

↳ abcdef

(2) find the Sum of odd Elements from the Array.

(3) Replace vowels with * in given string

(4) Replace the Sequence character by * Raiaghhhu

R*g***u

(5) WAP to search Element from the given Array. if found Reverse that number:

A = {10, 235, 384, 15}

B = {235, 16, 75, 15} ← compare

o/p

10 532 384 51

⑤ WAP to Replace the Special character by 1 in a given string.

① MIDDLE Element

```
class main1
{
 psvm (- - - -)
 {
  char[] ch = {'a','b','c','d','e'};
  String str = "abcde";
  char[] ch = str.toCharArray();
  for(int i=0; i< ch.length ; i++)
  {
   if ( i== ch.length%2 == 0)
   {
    SOP (ch[ch.length/2 -1] + ch[ch.length/2])
   }
   else
   {
    SOP (ch[ch.length/2])
}}}
```

② Sum of odd Elements

```
class main1
{
 psvm (- - - -)
 {
  char[] ch = {'1','2','3','4','6','6','7','8'}; int sum=0;
  for (int i=0; i< ch.length ; i++)
  { if (ch[i] %2 != 0)
    { sum = sum + ch[i];
    }
  }
 SOP("Sum of odd Elements: " + sum);
}
}
```

③ Replace vowels with *

```
String str = "YESWANTH";
char[] ch = str.toCharArray();
String res = "";
for(int i=0; i< ch.length ; i++)
{
 if (ch[i] == 'a' || ch[i] == 'A' ||
     'e' || 'E' || 'i' || 'I' || 'o' || 'O' || 'u' || 'U')
  SOP("*");
 else
  res = res + ch[i];
```

④ Replace Sequence characters by *

```
String str = "YEEESWAAAAANtthh
char[] ch = str.toCharArray();
String res = "";
for(int i=0; i< ch.length; i++)
{ int flag=0;
  while(ch[i]==ch[i+1])
  {
    flag=1; i++;
  }
  if(flag==1)
  {
    res = res+"*";
  }
  res = res + ch[i];
}
```

⑤ Search Element from Given Array If found reverse that num

```
for(i=0; i<arr2.length; i++)
{
for(j=0; j<arr1.length; j++)
  {
    if(arr2[i]==arr1[j])
    {     while(arr1[j]>0){
      int temp = arr1[j]%10;
          rev = rev*10+temp;
          arr1[j] = arr1[j]/10;
      }
      Sop(rev);
  }else{
  Sop(arr1[j]); }
}
}
```

~~for(i=0; i<arr2.length~~

25/5/16 (*) Display the Words initcap manner in the String

i/p ⇒ chitradurga is a float city

o/p ⇒ Chitradurga Is A Float City

i=0; ch[i-1];

```
Class main1
{
  psvm (------)
  {
    Scanner sc = new Scanner(System.in);
    Sop(" Enter Sentence: ");
    String str = sc.nextLine();
    char[] ch = str.toCharArray();
    String res = "";
    for(int i=0; i<ch.length; i++)
    {
      if (i==0 || ch[i-1] == ' ')
      {
        if (ch[i] >= 97 && ch[i] <= 122)
          res = res + (char)(ch[i] - 32);
        else
          res = res + ch[i];
      }
      else if (ch[i] >= 65 && ch[i] <= 90)
        res = res + (char)(ch[i] + 32);
      else
        res = res + ch[i];
    }
    Sop(res);
  }
}
```

(*) Reverse Words in a Sentense

0 1 2 3 4 5 6 7 8 9 10

I/P chitradurga is a fort city

O/P agrudartihc si a trof ytic

```
class main2
{
  psvm (- - - - - -)
   {
     Scanner sc = new Scanner (System.in);
     SOP ("Enter Sentence");
     String str = sc.nextLine();
     char[] ch = str.toCharArray();
     String res = "";
     for(int i=0; i< ch.length; i++)
      {
        int K=i;
        while (i< ch.length && ch[i]!= ' ')
         {
           i++;
         }
        int j= i-1;
        while (j>=K)
        {
          res = res+ ch[j];
          j--;
        }
       res = res +" ";
      }
   } SOP(res);
};
```

(*) Reverse Sentence

ip	chitradurga is a fort city
op	city fort a is chitradurga

```
class main1
{
  psvm (---)
  :{
     Sop ("Enter Sentence : ");
     String str = sc1.nextLine();
     char[] ch = str.toCharArray();
     String res = "";
    Sop ("After Reversing the Sentence");
   for (i = ch.length-1 ; i>=0; i--)
    {
      int k=i;
      while (i>=0 && ch[i] != ' ')
      {
        i--;
      }
      int j=i+1;
      while (j<=k)
      {
        res = res + ch[j];
        j++;
      }
      res = res + " ";
    }
  } Sop (res);
}
```

(*) Reverse the Specified word from the String

If chitradurga is a fort city

Enter World to be Search ⇒ fort

or

fort is present

After Reversing ⇒ chitradurga is trof city

```
class main
{
psvm (- - - - -)
{
SOP ("Enter Sentence");
String str = sc.nextLine();
char[] ch = str.toCharArray();
String res = "";
int k, L, count=0;
for(int i=0; i<ch.length; i++)
{
int j=0;
While (i<ch.length && j<ch1.length && ch[i]==ch[j])
{
if (count==0)
k=i;
res = res + ch[i];
i++;
j++;
}
L=j-1;
if (j==ch1.length && ch[i]==' ')
{
SOP(str1 + "is present");
SOP("After Reversing");
while (L>=k)
{ res = res + ch[L];
} } L--;
```

```
SOP("Enter word to be search");
Strip str1 = sc.nextLine();
char[] ch1 = str1.toCharArray();
```

```
        res = res + ch[i];
    }
    SOP(res);
}
}
```

(*) Given two Strings are Anagram or not

Ex:-

(1) Keep
(2) peek
} ⇒ these two Strings are Anagram

Ex:-

(1) Keep
(2) pek
} ⇒ NOT Anagram

Step 1 :- Removal of Spaces
Step 2 :- No. of charecters
Step 3 :- Both the strings set to one case
Step 4 :- Sort 2 strings
Step 5 :- Compare

(1) keep listen
(2) peek silent
→ (1) → keeplisten, peekSilent → (2) → 10, 10 → (3) → keep listen, peek silent → (4) → eeeiklnpst

Anagram ← (5) ← eeeiklnpst, eeeiklnpst

```
import java.util.Scanner;
class Anagram
{
 psvm(- - - ;- -)
 {
  Scanner sc = new Scanner(System.in);
  Sop("Enter String1"); String st1 = sc.nextLine();
  Sop("Enter String2"); String st2 = sc.nextLine();
    st1 = removeSpace(st1);
    st2 = removeSpace(st2);
  if(st1.length() != st2.length())
  {
   Sop("Strings are not an Anagram");
  }
  else
  {
    st1 = toLower(st1);
    st2 = toLower(st2);
    st1 = Sort(st1);
    st2 = Sort(st2);
```

method ①

```
if(st1.equals(st2))
{
 Sop("Strings are an Anagram");
}
else
{
 Sop("Strings are not an Anagram");
}
```

Pribuilt method

(or)

method ②

```
boolean r = Compare(st1, st2);
if(r == true){
Sop("Strings are an Anagram");
else
 Sop("Strings are not an Anagram");
}
}
```

User defined method

```
static String removespace(String st)
{
    String res = "";
    char ch[] = st.toCharArray();
    for(int i=0; i<ch.length; i++)
    {
        if(ch[i] != ' ')
        {
            res = res + ch[i];
        }
    }
    return res;
}
static String toLower(String st)
{
    String res = "";
    char ch[] = st.toCharArray();
    for(int i=0; i<ch.length; i++)
    {
        if(ch[i] >= 65 && ch[i] <= 90)
            res = res + (char)(ch[i]+32);
        else
            res = res + ch[i];
    }
    return res;
}
static String sort(String st)
{
    char ch[] = st.toCharArray();
    for(int i=0; i<ch.length; i++)
    {
        for(int j=i+1; j<ch.length; j++)
        {
            if(ch[i] > ch[j])
            {
                char temp = ch[i];
                ch[i] = ch[j];
                ch[j] = temp;
            }
        }
    }
    return new String(ch);
```

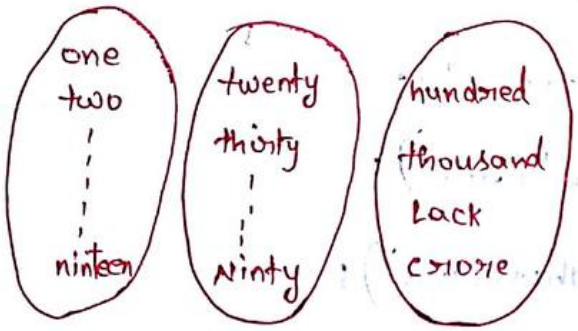
```
static boolean Compare (String st1, String st2)
{
  char ch1[] = st1.toCharArray();
  char ch2[] = st2.toCharArray();

  for (int i=0; i< ch1.length; i++)
  {
    if (ch1[i] != ch2[i])
      return false;
  }
  return true;
}
}
```

27/5/16

1 Display the Number in the form of Sentence.

i/p → 5927

o/p → Five thousand Nine hundred twenty Seven



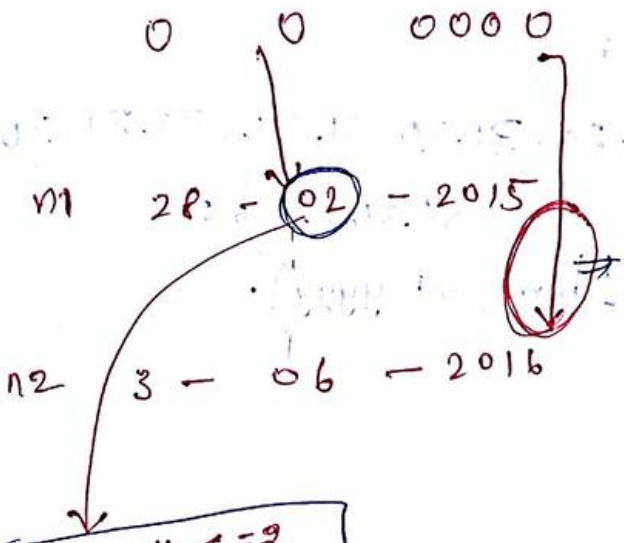
⟹ These are all unique words.

```
class main1
{
  String one[] = {"", "ONE", "TWO", "Three", "Four", "Five",
    "Six", "Seven", "Eight", "nine", "Ten", "Eleven",
    "twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen",
    "Seventeen", "Eighteen", "Nineteen"};
```

```
String two[] = {"", "", "Twenty", "Thirty", "Fourty",
                "Fifty", "Sixty", "Seventy", "Eighty",
                "Ninety"};

static void pw (int n, String st)
{
    if (n > 19)
        Sop (two[n/10] + one[n%10]);
    else
        Sop (one[n]);
    if (n != 0)
        Sop (st + " ");
}
Psvm (- - - - - - - - - -)
{
    Scanner sc = new Scanner (System.in);
    Sop ("Enter the Number");
    int n = sc.nextInt();
    pw (n/10000000, "Crore");
    pw ((n/100000)%100, "Lakh");
    pw ((n/1000)%100, "Thousand");
    pw ((n/100)%10, "Hundred");
    pw (n%100, "");
}
```

(*) Count the Number of days between the Dates

Ex: 28-2-2015 to 3-6-2016



(1) if month <= 2

$$\left(2014/4 - 2014/100 + 2014/400\right) + 28 + m[0] + 2015 * 365$$

(No. of Leap years) + (current month days completed) + (previous month) + (no. of days)

(2) if month > 2

$$\left(2015/4 - 2015/100 + 2015/400\right) + \text{current month days completed} + \text{Previous months} + 2015 * 365$$

(*) Leap year ⇒

number /4 & /400 not /100

num/4 && num/400 num!/100

$$\frac{2015}{4} + \frac{2015}{400} - \frac{2015}{100} \Leftarrow$$

(*) Count the Number of Days Between the DATES

```
public class Date
{
   int dd, mm, yyyy;
   final int m[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30
                     31, 30, 31 };
   public Date (int dd, int mm, int yyyy).
   {
     this.dd = dd;
     this.mm = mm;
     this.yyyy = yyyy;
   }
   public int numberOfDays ()
   {
     int days = yyyy * 365;
     for(int i=0; i<mm-1; i++)
     {
       days += m[i];
     }
     if (mm > 2)
       days += (yyyy)/4 - (yyyy)/100 + (yyyy)/400;
     else
       days += (yyyy-1)/4 - (yyyy-1)/100 + (yyyy-1)/400;
     return days + dd;
   }
   public String toString()
   {
     return "" + dd + "." + mm + "." + yyyy + " ";
   }
```

```
public class main2
{
 static Date readDate()
 {
  Scanner sc = new Scanner(System.in);
  Sop("Enter the dd");
  int dd = sc.nextInt();
  Sop("Enter the mm");
  int mm = sc.nextInt();
  Sop("Enter the yyyy");
  int yyyy = sc.nextInt();
  return new Date(dd, mm, yyyy)
 }
public static void main(String[] args)
 {
  Date d1 = readDate();
  Date d2 = readDate();

  int count1 = d1.numberofDays();
  int count2 = d2.numberofDays();
   if(count1.> count2)
   {
    Sop("number of days b/n " + d1 + "And " +
                d2 + " : " + (count1 - count2));
   }
   else
   {
   Sop("NO of Days b/n " + d1 + "And" + d2 +
          " : " + (count2 - count1));
   }
 }
}
```

(*) Sub-String

```
import java.util.Scanner;
class main
{
  static boolean check(String str, String st)
  {
    char ch1[]= str.toCharArray();
    char ch2[]= st.toCharArray();
    int count=0;
    for(int i=0; i<ch1.length; i++)
    {
      int j=0;
      int k=i;
      while(j<ch2.length && i<ch1.length &&
          ch1[i]==ch2[j] || ch1[i]==ch2[j]+32 ||
              ch1[i]==ch2[j]-32)
      {
        i++;
        j++;
      }
      if(j==ch2.length)
      {
        count++;
        S.O.P(count+" time pos:"+k);
      }
      i=k;
    }
    if(count>0)
    { SOP(st+" occured "+count+"times");
    } return true;
```

```
    return false;
}
public static void main (String [] args)
{
Scanner sc = new Scanner (System.in);
SOP("Enter main String");
String str = sc.nextLine();
SOP("Enter Substring which one to be check");
String st = sc.nextLine();
boolean rs = check (str, st)
   if (rs)
   {
    SOP (st + " present in the " + str);
   }
   else
   {
    SOP (st + " not present in the " + str);
   }
}
}
```

i/p Enter String

Hi How are you, How are you doing ?

Enter Substring

How

1 time pos : 3

2 time pos : 15

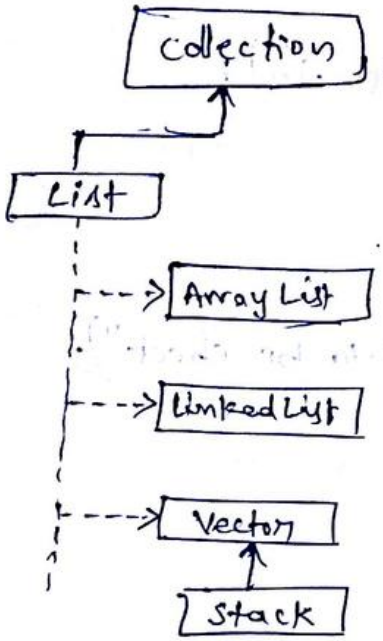
How occured 2 times

How present in the Str.

your

your not present in the String

(*) b Implement your own ArrayList Given by a name · Raghulist



(*) What are the prop of List?

(1) duplicates are allowed
(2) more than one Null insertions Allowed (*)
(3) Heterogenous.
(4) Insertion order maintain (or) preserved

Array List :- growable (or) Resizable Array

deletion (or) insert an Elements from Array [By using Left and Right shifts]

When it is used ? TO Retrieve.

When it is not used ? Insert & Deletion

Linked List :- doubly linked List

Data Structure :- organizing the memory for data is called Data Structure

When it is used ? for Insertion & Deletion

When it is not used :- When Ever frequent operation is Retreival

(*) ArrayList having "Overload Constructor"

`javap java.util.ArrayList` ⟷ This will give Entire info about ArrayList class

types of Constructors

`ArrayList arr = new ArrayList();` → Defaultly creates array with 10

`ArrayList arr = new ArrayList(8);` → It creates array with size 8

`ArrayList arr1 = new ArrayList(arr);`

✳ `collection c = new ArrayList();`

Here we are Casting ArrayList to collection. So we can use only properties of collection Interface.

- add
- addAll
- removeAll
- retainAll
- clear
- isEmpty
- Size()
- toArray()
- iterator()

(*) Implement your own ArrayList given by a name RaghuList

(or) JList

```
package first;
import java.util.collection;
public class JList
{
   private object ob[];
   private int i=0;
   private int cpy;
   public JList()
   {
      cpy=10;
      ob.= new object[cpy];
   }
   public JList(int cap)
   {
      cpy = cap;
      ob = new object[cpy];
   }
   public boolean add(object obj)
   {
      if (i < ob.length)
      {
         ob[i] = obj;
         i++;
         return true;
      }
      else
      {
         cpy = cpy * 3/2 + 1;
         object nob[] = new object[cpy];
```

```
        for(int i=0; i< ob.length; i++)
        {
          nob[i] = ob[i];
        }
        nob[i] = obj;
        ob = nob;
        i++;
        return true;
    }
}
public boolean addAll(Collection c1)
{
  Object obj[] = c1.toArray();
  for(int i=0; i< obj.length; i++)
  {
    add(obj[i]);
  }
  return true;
}
public int Size()
{
  return i;
}
public int Capacity()
{
  return cpy;
}
public boolean isempty()
{
  if(i==0) {return true;}
} else {return false;}
public void clear()
{
  ob = new object[cpy];
  i=0; → for start index from 0
}
```

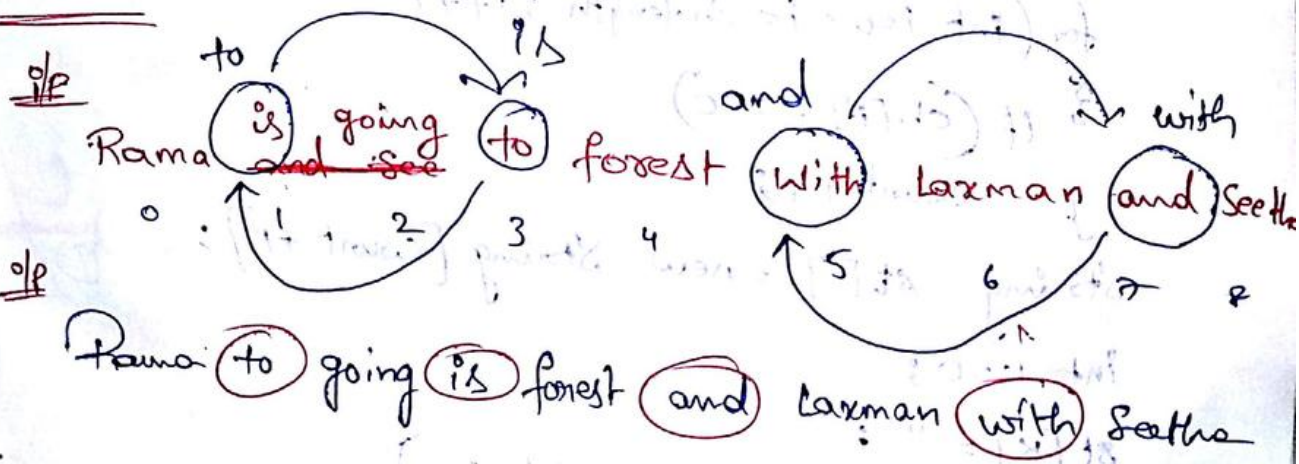
```
        if(obj[i].equals(ob[j]))
        {
          remove(ob[j]);
        }
      }
    }
    return true;
}
```

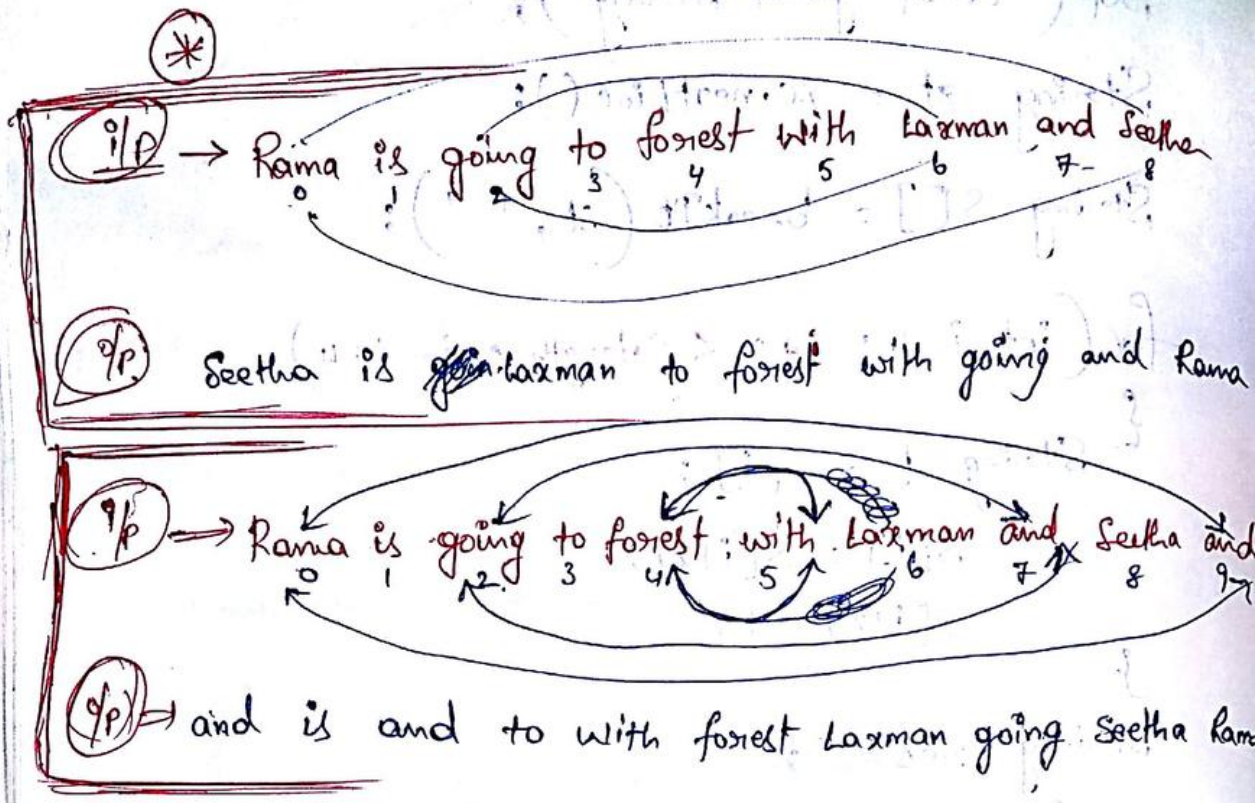
⊛ Split method (User-Defined).

the words and Interchange odd positions in String

```
public class main1
{
  static String[] breakIt(String str, char c)
  {
    char ch[] = str.toCharArray();
    int count = 0;
    for(int i=0; i<ch.length; i++)
    {
      if(ch[i] == c)
    }     count++;
    String st[] = new String[count+1];
    int k=0;
    st[k] = "";
    for(int i=0; i<ch.length; i++)
    {
      if(ch[i] != c)
        st[k] = st[k] + ch[i];
      else
      {
        k++;
        st[k] = "";
      }
    }
    return st;
  }
  public static void main(String[] args)
  {
    Scanner sc1 = new Scanner(System.in);
```

```
Sop("Enter your String");
String st = sc.nextLine();
String s[] = breakIt(st, ' ');
for(int i=1 ; i+2 < s.length ; i= i+4)
 {
   String t = s[i];
        s[i] = s[i+2];
        s[i+2] = t;
 }
String res = "";
for(int i=0 ; i < s.length ; i++)
 {
  res = res + s[i] + " ";
 }
 Sop(res);
 }
}
```






```
public class main1
{
 static String[] breakIt(String str, char c)
  {
    char ch[] = str.toCharArray();
    int count =0;
    for(int i=0; i< ch.length ; i++)
    { if(ch[i] == c)
      count++;
    String st[] = new String [count+1];
    int k=0;
    st[k] = "";
    for(int i=0 ; i < ch.length; i++)
    { if(ch[i] != c)
      st[k] = st[k] +ch[i];
      else
    { k++; st[k] = ""; }
 return st;
```

```
public static void main (String[] args)
{
 Scanner sc1 = new Scanner (System.in);
  Sop ("Enter you String");
  String st = sc1.nextLine();
  String s[] = breakIt (st, ' ');
  if(s.length%2 ==0)
 {for(int i=0; i< s.length/3-1; i+=2)
   {
     String t = s[i];
          s[i] = s[s.length-i-1];
        s[s.length-i-1] = t;
 }}
  String res = "";
  for(int i=0; i< s.length ; i++)
  {
    res = res + s[i] + " ";
  }
  sop (res);
 }
}
for if(s.length%.
  else
  {
   for(int i=0; i< s.length/4 ; i+=2)
    {
      String t = s[i];
          s[i] = s[s.length-i-1];
     s[s.length-i-1]= t;
 }}
```

(*) ~~Find the~~ Display the middle character of middle word in a Sentence

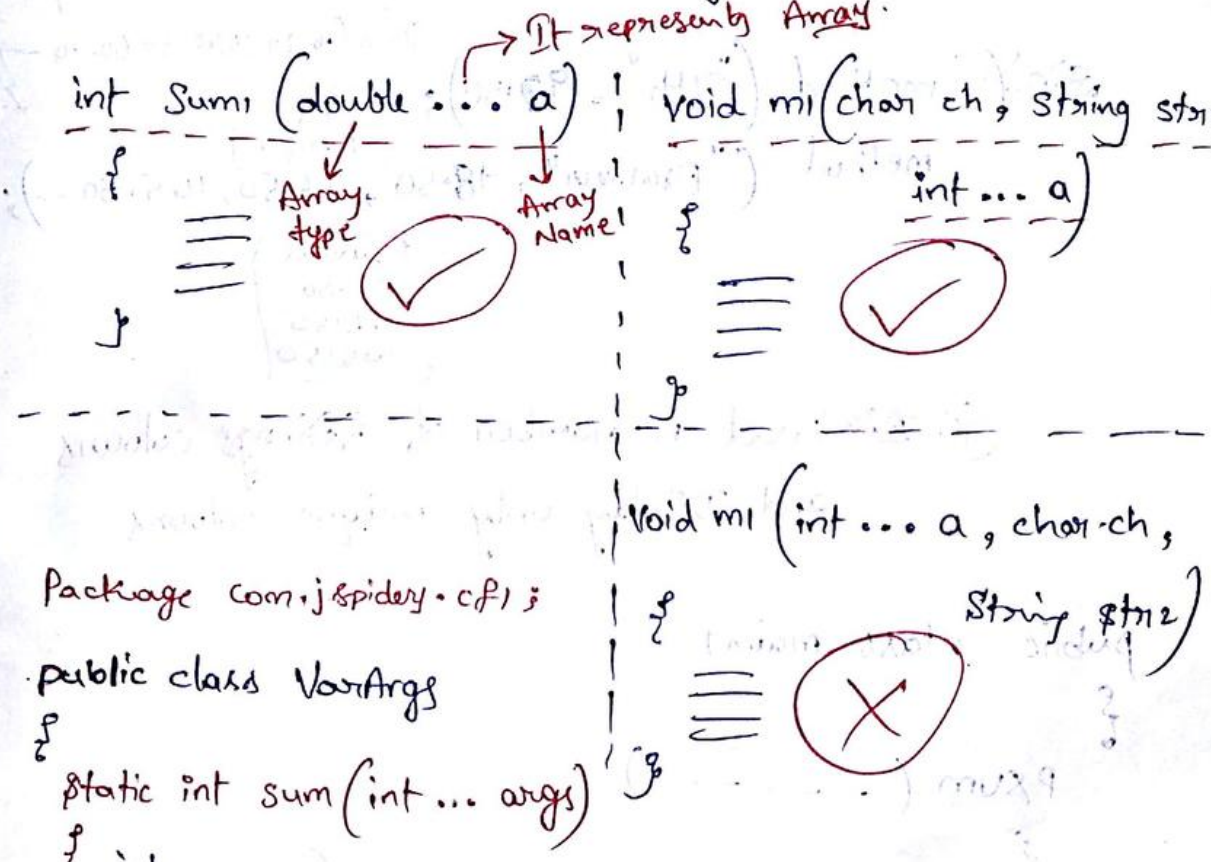
i/p → hai how are
o/p → o

i/p hai how a reyon
o/p ow

```
public class Main
{
  P s vm (String[] args)
  {
    Scanner scn1 = new Scanner (System.in)
    SOP("Enter string");
    String st = scn1.nextLine();
    String s[] = breakIt(st, ' '); last program method
    for(int i=0; i<s.length/2; i++)
    {
      if(i == s.length/2)
      {
        char ch[] = s[i].~~length~~ toCharArray();
        if(ch.length %2 ==0)
          SOP(ch[ch.length/2 -1] + "" ch[ch.length/2])
        else
          SOP(ch[ch.length/2]);
      }
    }
  }
}
```

Varargs

* If Any method having arguments of same type differ by no of parameters then we go for Varargs
* It is useful to decrease the length of the program
* method Signature can have only one Varargs and it should be last Argument.




```
Package com.jspiders.cfl;
public class VarArgs
{
 static int sum(int ... args)
 {
   int s=0;
   for(int i=0; i< args.length; i++)
     s = s + args[i];
   return s;
 }
```

```
static void method(String st, double b, char...ch)
{
  Sop(st);
  Sop(b);
  for(int i=0; i<ch.length; i++)
     Sop(ch[i]);
}

psvm(----)
{
  Sop("* * * * * * *");
  Sop("Sum of int Elements" + sum(10));
  Sop("Sum of int Elements" + Sum(10,20,30,40));
  method("SITh", 97.50);
  method("praveen", 98.50, 74.50, 105.50);
```

o/p
10
100
Sum(10,20,30,40,50,60,70)

SITh
97.50

praveen
98.50
74.50
105.50

(*) Read n number of colours and Display only unique colours

```
public class main1
{
  psum(------)
  {
    Scanner scn1 = new Scanner(System.in);
    String[] st =
    Sop("Enter num of colours");
    int n = scn1.nextInt();
```

```
String str[] = new String[n];
for(i=0; i<n; i++)
  str[i] = sch1.next();
int count;
for( i=0; i<n; i++)
{  count=0;
  for(j=i+1; j<n; j++)
    {  if(str[i].equals(str[j]))
      {  count++;
        for(int k=j; k<n-1; k++)
          {  str[k-1] = str[k];
          }
        n--;
        j--;
      }
    }
  if(count==0) { SOP(str[i]); }
}
```

~~for(int i=0; i<n; i++)~~
~~{ SOP(str[i]);~~
}

```
Public static String[] colours (int n, String[] ...str);
{ int count;
  for(int i=0; i<n-1; i++)
    count=0;
    for(j=i+1; j<n; j++)
     {  if(str[i].equals(str[j]))
       {  count++;
         for(int k=j; k<n-1; k++)
           str[k-1] = str[k];
         n--;
         j--;
       }
     }
  if(count==0)
    SOP(str[i]);
}
```