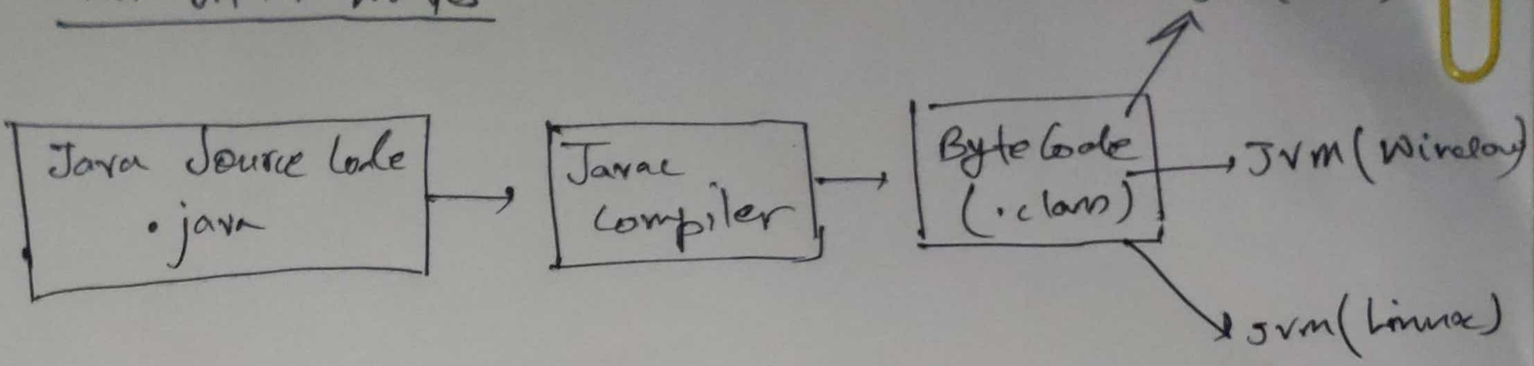
How JAVA WORKS



→ Hello world
→ DT
→ Post/Past Increment
→ Type Casting.

JAVA R1

~~OOPS TOPICS~~

- Hello World

```
public class HelloWorld{
        public static void main(String[] args)
        { System.out.println ("Hello world");
        }
}
```

- No gap between class name
  ⇓
  CamelCase convention

Eg.
rateOfInterest
small (lowercase) — big. (uppercase).

Also case sensitive

System, String, println ✓

System, string, Println } ✗

Data Types

① byte 8bits ② long 8 bytes ③ int 4 byte ④ short 2 byte. ⑤ double ⑥ boolean ⑦ char ⑧ float

Post | Past Increment

```
b = 45
d = b++
System.out.println(b) = 46
                  (d) = 45
```

```
b = 45
d = ++b
System.out.println(b) = 46
                  (d) = 46
```

TYPE CASTING : Assign a value of one primitive data type to another.

Automatic Type Casting (ATC)

$$\begin{tikzcd} \text{byte} \arrow[r] & \text{short} \arrow[r] & \text{char} \arrow[r] & \text{int} \arrow[r] & \text{long} \arrow[r] & \text{float} \arrow[r] & \text{double} \end{tikzcd}$$

← doing reverse
: Manual Type Casting

Eg.

```
int a = 4
long b = a
print (b)
```

4

↳ following ATC

But if

```
double p = 3.14
int q = p        → This line will give error
print (a)
```

↳ Reverse of ATC

∴ manual TC is required

Soln:

```
double p = 3.14
int q = (int) p
print(a)
```

✓

3

(3) Scanner function in Java. (5)

```
import java. util. Scanner
public class SimpleInterest {
  public static void main (String [] g>)
  { Scanner sc = new Scanner( System.in)
       int principal = 500          = sc.nextInt();
       float rate = 12.5f           = sc. nextFloat();
       int time = 12                = sc.nextInt();

    float SI = principal × rate × float /100;
    System.out.println ( "the SI is" + ___ );
    }
}
```

(Pb) Take input of a string after a Number.

```
{ int num = sc . next Int();
   sc.next Line();
   String str = sc. next Line();
  System . out. println(num);
  System . out. println(str);
}.
```

→ Operators   5 types.

(1) Airthematic	+, -, ×, /, %, ++, --
(2) Bitwise	NOT, AND, OR, XOR
(3) Assignment	=, +=, -=, /=, %=, ×=, >>=, <<=, &=, |=
(4) Comparison	==, !=, >, <, >=, <=
(5) Logical	&&, ||, !

(4)

(6)

NOT opposite	
0	1
1	0

		AND Both 1 =1	OR atleast 1	XOR only one 1
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Right shift
÷2

1101 >>

a = 13
13 >> 1

1 bit Right shift

1 1 0 1
→ → → 0 1
1

1 1 0 1
↘ ↘ ↘ 0
0 1 1 = 6

Left shift
×2

a = 1 1 0 1 = 13
1 1 0 1 0 ⇒ 26

int a = 13
b = 13 >> 1 6
c = 13 >> 2 $\frac{13}{2^2}$ = 3
d = 13 >> 3 $\frac{13}{2^3}$ = 1

a = 13
b = 13 << 1 13 × 2^1 = 26
c = 13 << 2 13 × 2^2 = 52
d = 13 << 2 13 × 2^3 = 104

13
8
104

Trick.

1101.0000 ... >> 1 ⇒ 110.000
110 6

Right shift by x ⇒ Eliminate last x bits from last
÷ 2^x : Dec form,

1101.0000 ... << 1 ⇒ 11010.000
= 26

Left shift by x ⇒ Add x zeros in binary form
× 2^x : Dec form

Right shift by x ⇒ $/2^x$ in Decimal.

eliminate last x bits in Binary

Left shift by x ⇒ $\times 2^x$ in decimal

~~at~~ append x bits of zeros in binary form

Q4. Shorthand (Ternary Op)

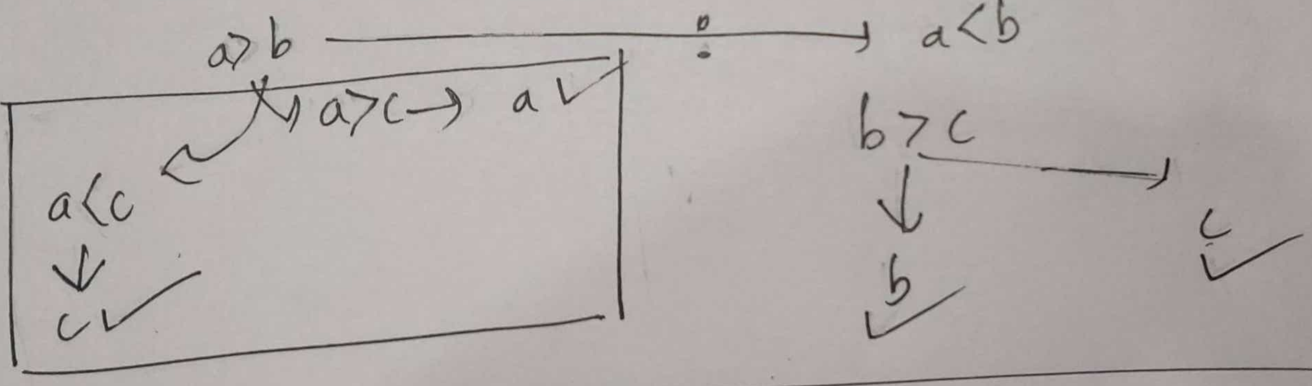
variable (cond) ? exp1 : exp2.

Greatest of three numbers.

→

result = a > b ? a > c ? a : c : b > c ? b : c ;

Explanation. $\boxed{a > b ? \; a > c \; ? \; a : c} : \boxed{b > c ? b : c}$

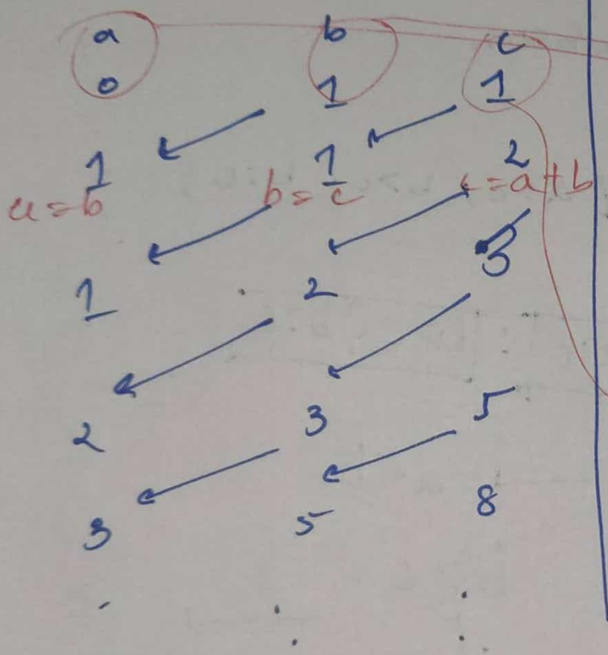


Switch Case

(P6) factorial.

```
int n=5                          5!.
for(i=n ; i>0; i--)
   { product = product*n };
 print(product)
```

(P7) fibonacci.



1st 7 num.

0 1 1 2 3 5 8 - ~

```
int n = 7.
 int a=0, b=1
int c,i

for(i=0; i<n-2; i++)
  { c=a+b
    print (c)
    a=b
    b=c
  }
```

(P8) Prime OR NOT.

~~If a num is prime, then its atleast on factor wi~~

For any number (n), atleast one of its factor will be present in (2, √n] i.e before =√n.

i*i=n

```
        bool p= ~~false~~ true;
  for(i=2; ~~i≤√~~ i*i<=n ; i++)
      { if (n%i ==0)
           { ~~p=true;~~
             p=false;
         break; }
   if(p) print ("Num is prime");
   else print ("Num is not prime");
```

(P9)

Series Sum $1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots \frac{1}{n}$ Ans

Input = n   o/p = Ans.

```
int nums n; sum = 0;
for (i = 1; i<=n; i++)
  {sum = sum + 1/i};
  print (sum);
```

(P10) $1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} + \dots \frac{1}{n}$

I/p = n   o/p = Ans.

```
int n;  sum = 0
{
 for(i=1; i<=n; i++)
 { if (i%2 != 0)
     { sum = sum + 1/i;}
     ~~elseif~~
     else { sum = sum - 1/i;}
 }
  print (sum);
```

	If ∞ loop or not
for (i = 0; i<10; i++)	Y
for (i = 0; ; i++)	Y
for (; ; i++)	N : i not declared, what to t
for (; ;)	Y

continue , break

(P11) Run a loop until the user input -ve no

```
{
 Scanner sc = new Scanner (System.in)
  { for(;;)
      { int  n = sc.nextInt();
             if(n<0)
               break;
      }
  }
```

Print the pattern(1).

P12.

```
*
* *
*  *  *
*  *  *     *
```

else print (Num is not Prime).

P12 P2

```
   4
i= 1  x
      x x
   2  x x x
   3  x x x x
i= 4
   n
```

```
int i, j;
for( i=1 ; i<=4 ; i++)                          "
{ for ( j=1; j <= i; j++)
  { System.out.print( ("x ");
  } System.out.println();
}.
```

(4)

P13

```
~~x x x x~~ 1
--~~x x x~~ 2
----~~x x~~ 3
      ~~x~~ 4
```

P3

```
for(i=1 ; i<=n; i++)
{ for (j=1 ; j < 2(i-1); j++)
  { System.out.print ("<onespace>");
  }
for( int j=1 ; j < n-i+1 ; j++)
  { System.out.print ("x-");
  }
  System.out.println();
]
```

space		start
i=1	0	4 (4-1+1)
2	2	3 (4-2+1)
3	4	2
n=4	6	1
	a+(n-1)d	n-i+1

$0+(i-1)2$

$\Rightarrow 2(i-1)$ spaces..

P14 P4

```
x x x x
x x x
x x
x
```

1	4
2	3
3	2
4	1

```
for(i=1; i<=n; i++)
{
   for( j=1 ; j<= (n-i+1); j++)
   { print ("x-"); }
   println()
}.
```

P15

```
      x
     x x
    x x x
   x x x x
```

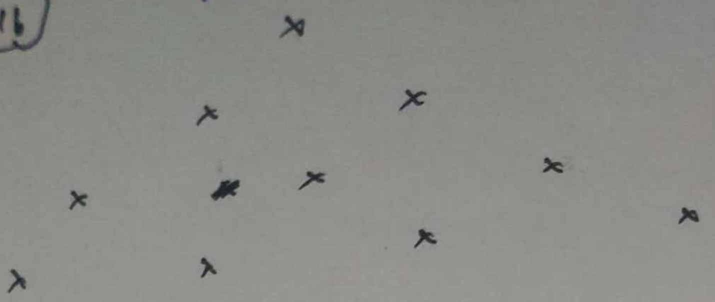
	*	spaces
i= 1	1	6
2	2	4
3	3	2
4	4	0

Consider two spaces as one

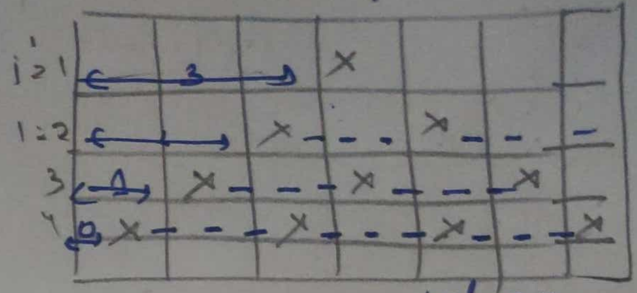
--

```
for(i=1 ; i<=n ; i++)
{
  for(j=1; i<n-i ; j++)        i=1   3
  { print("--");                 2   2
  }                              3   1
  for(j=1 ; j <= i ; j++)        4   0
     { print ("*-");
     } println();
}.
```

P16 P5



12



	~~stars~~ with spaces	~~2x~~ spaces before x
i=1	1 - - -	3
2	2 - - -	2
3	3 - - -	1
4	4 - - -	0
		n-i

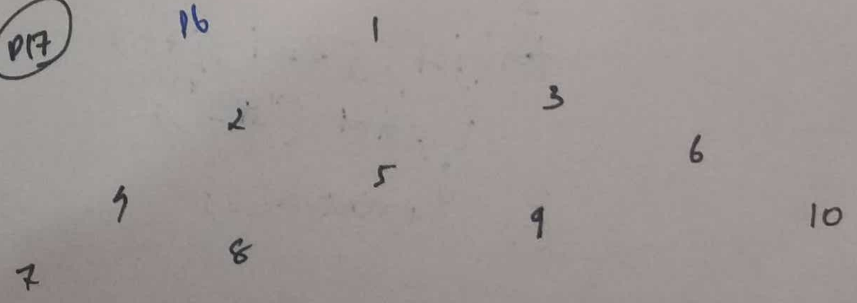
```
for(i=1 ; i<=n ; i++)
  {for(j=1 ; j<=n-i; j++)
   { print("--"); }

  for(j=1 ; j<=i; j++)
     print("x---");
   printLn();
  }.
```

After every x, 3 space gap in present

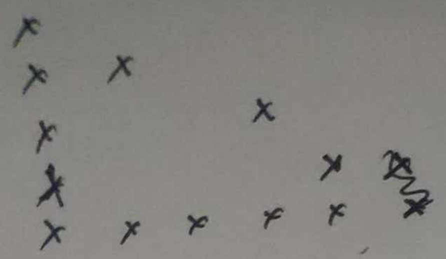
	~~starts~~ with 3 spaces	2 ~~x~~ spaces before x
i=1	1 - - -	3
2	2 - - -	2
3	3 - - -	1
4	4 - - -	0
i=1 ; i<n	~~i~~ j=1 j<i	n-1

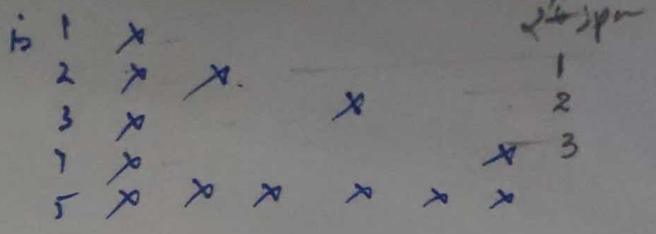
At ith row, $2^x(n-i)$ spaces are present
In every row, after every x, 2 spaces are present.

P17 P16



PC9





Every row has 2 stars except last one.

```
* for(i=1 ; i<=n ; i++)
        {.  if (i==1 && i==n)
           { for( j=1; j< i ; j++)
                 print("* - ");
           }
             else.
               {  print("* - ");
                  for (int j=1;  j<=(n-i); j++)
                     {print(" - - ");}
                  print("*")
               }.
           }.
```

Hint: 1st & last row has same no. of stars as the row number
Rest of the rows has two stars separated by 2*(n-1) spaces.

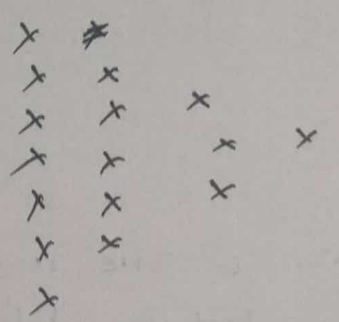
Almost same;

just at every row instead of x, we have make a variable int number=1 & print number at x & number++.

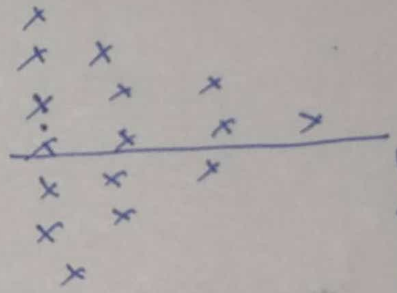
```
int number=1
for( int i=1 ; i<=n ;i++)
   { for (int j=1 ; j<=n-i ; j++)
        { print ("--"); }

     for (int j=1; j<=i ;j++)
        { print(number + "---"); }
              number++;
     ~~S~~ println();

   }.
```

P18   n = 4

p7



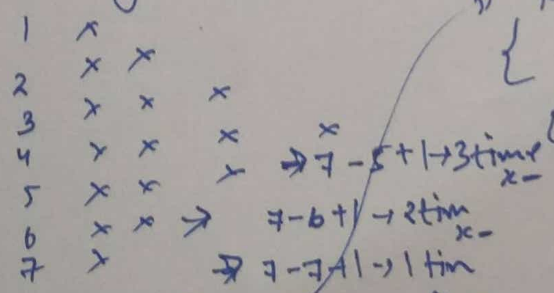
1 way.



we know for n=4 (P1)

n=3 (P4)

2nd way.



rows = 2n-1

```
for(int i=0; i<=rows; i++)
{       if(i<=n)
  {-for(int j=1; j<=i ;j++)
        print("x -");
  }

  else
  { for(int j=1; j<rows-i+1; j++)
        {print("x-");}
  }
  println();

}
```