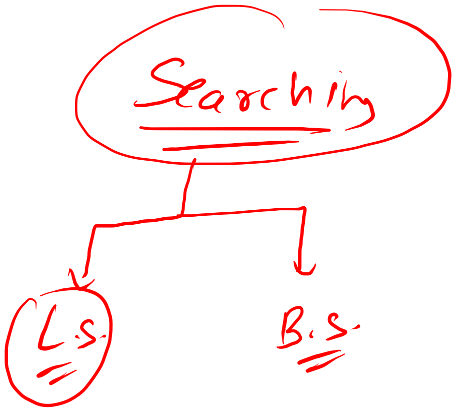
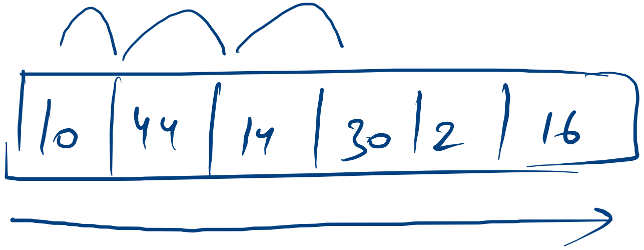


Find



x = 30

Binary Search

↳ Data must be sorted

$X = 120$

						lo		mid			hi
0	1	2	3	4	5	6	7	8	9	10	11
10	20	25	36	42	75	83	120	150	216	217	318

itr	lo	hi	mid
1	0	11	5
2	6	11	8
3	6	7	6
4	7	7	7 → 7

		Search A.		Itration	operat
$\frac{n}{2^0}$	←	n	→	1	K
$\frac{n}{2^1}$	←	$\frac{n}{2}$	→	1	K
$\frac{n}{2^2}$	←	$\frac{n}{4}$	→	1	K
		⋮		⋮	
$\frac{n}{2^{m-1}}$	←	1		1	K

m steps

$$T.O. = \underbrace{K+K+K+K \dots + K}_{m \text{ times}} \quad — (I)$$

$$\Rightarrow \frac{n}{2^{m-1}} = 1$$

$$\Rightarrow n = 2^{m-1}$$

⇒ taking log both sides

$$\Rightarrow \log_2 n = m-1$$

$$m = \log_2 n + 1 \quad — (II)$$

$$T.O. = m.K$$

$$T.O. = (\log_2 n + 1) K$$

$$T.O. \propto \log_2 n \Rightarrow O(\log_2 n)$$

$$\underbrace{10+10+10} \Rightarrow (10 \times 3)$$

if(x == arr[mid]){ ceil = floor = arr[mid]};

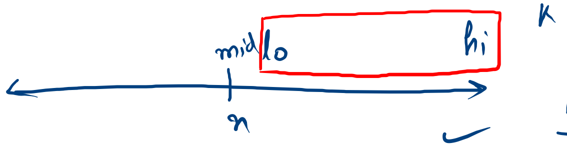
k = 3

arr =

1	5	10	15	22	33	40	42	55	66
0	1	2	3	4	5	6	7	8	9

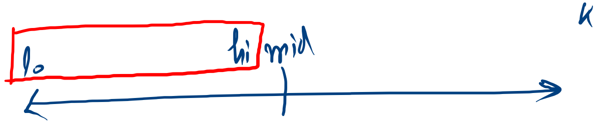
✓ ceil => just larger

✓ floor => just smaller



k > arr[mid]

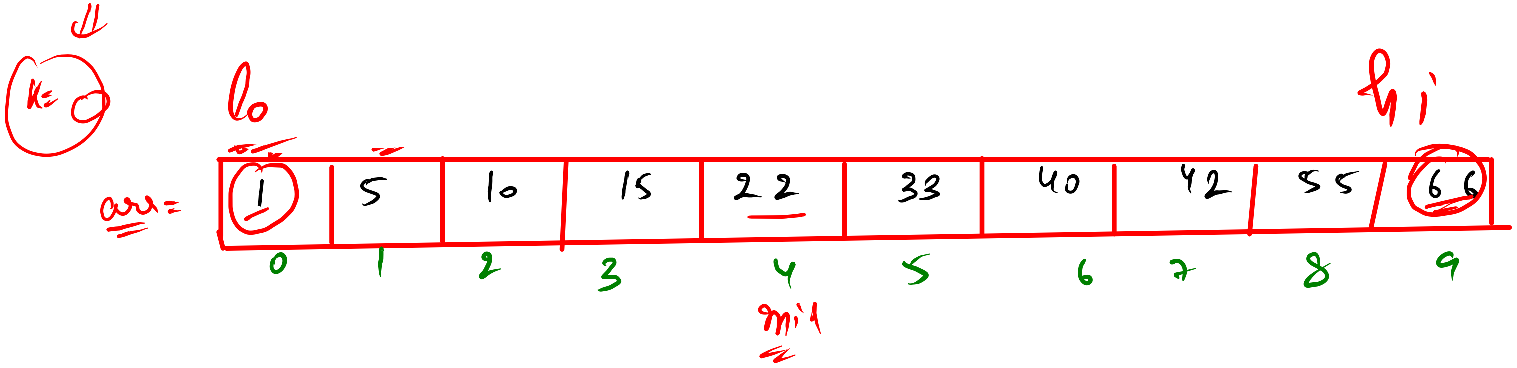
floor = arr[mid]



k < arr[mid]

ceil = arr[mid

K = 34



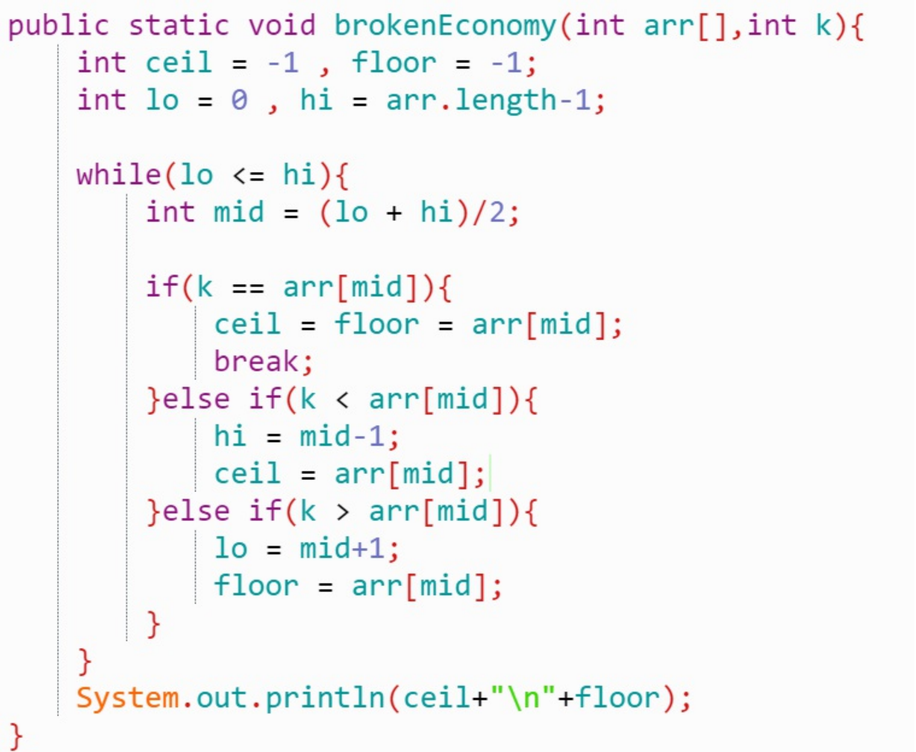
22

Ceil = ~~-1~~ 22

floor = ~~-1~~ 22

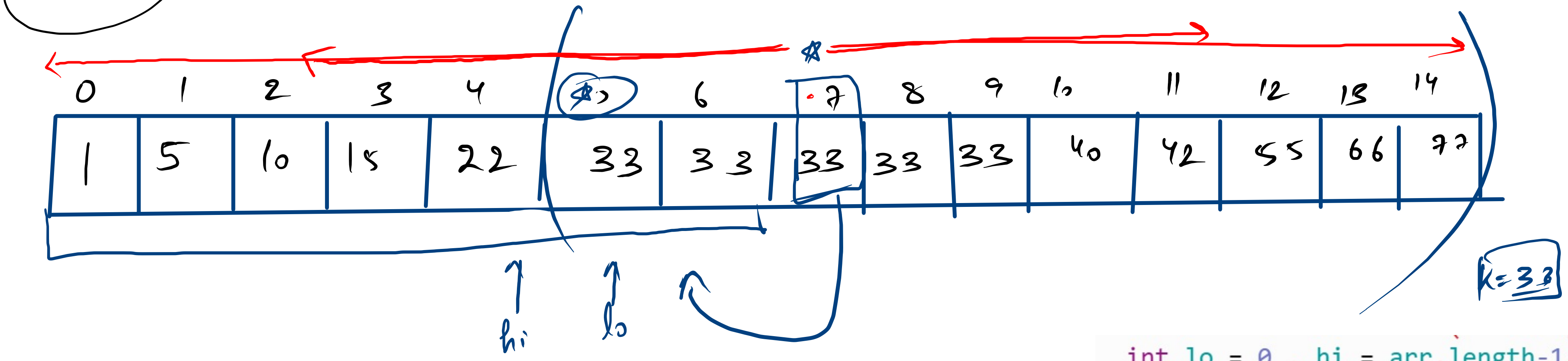
lo = 0

hi = 9

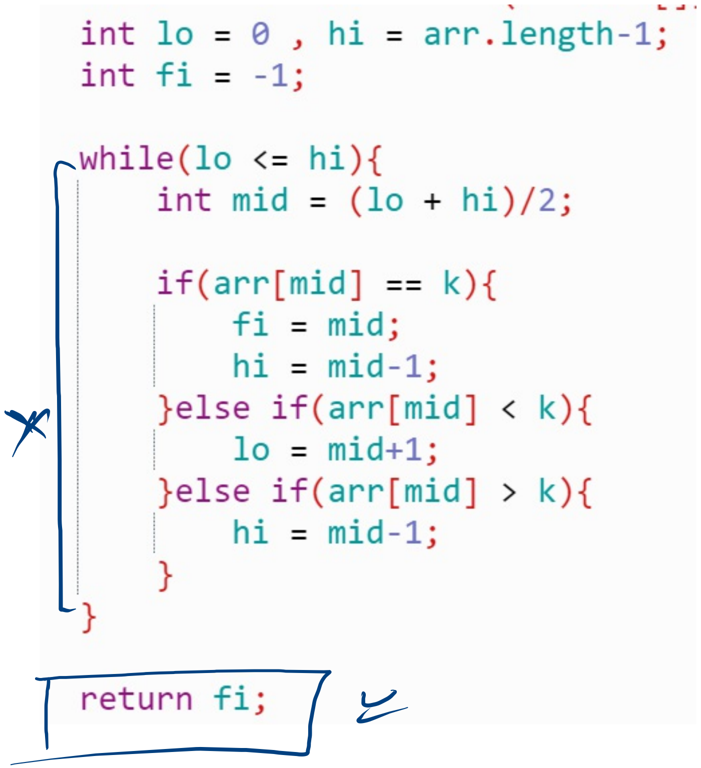


```
public static void brokenEconomy(int arr[],int k){
    int ceil = -1 , floor = -1;
    int lo = 0 , hi = arr.length-1;

    while(lo <= hi){
        int mid = (lo + hi)/2;

        if(k == arr[mid]){
            ceil = floor = arr[mid];
            break;
        }else if(k < arr[mid]){
            hi = mid-1;
            ceil = arr[mid];
        }else if(k > arr[mid]){
            lo = mid+1;
            floor = arr[mid];
        }
    }
    System.out.println(ceil+"\n"+floor);
}
```

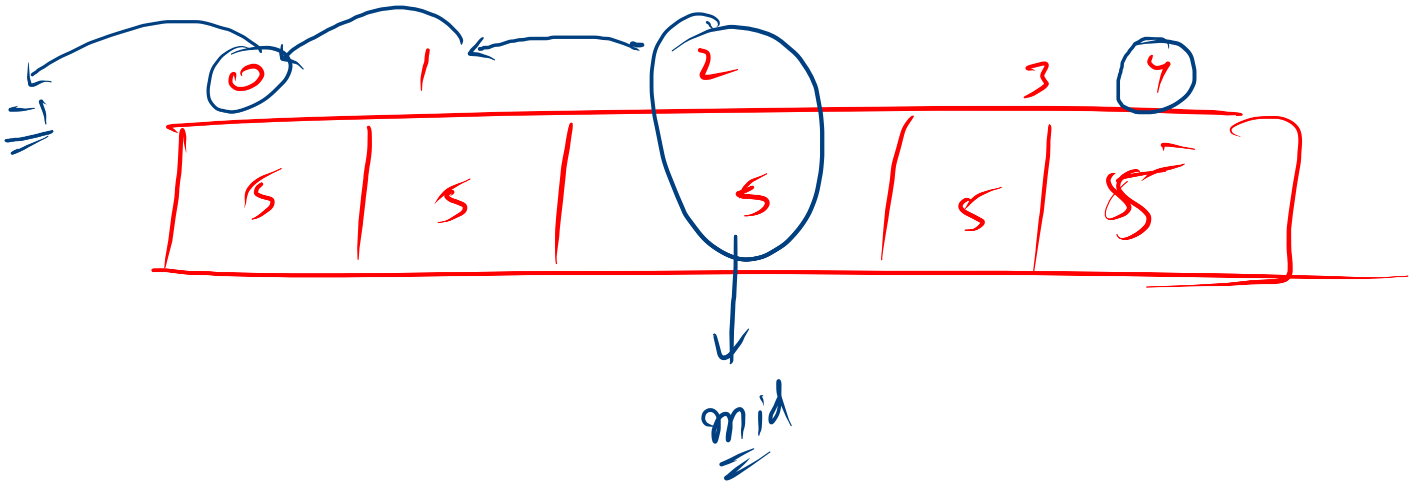
k=33



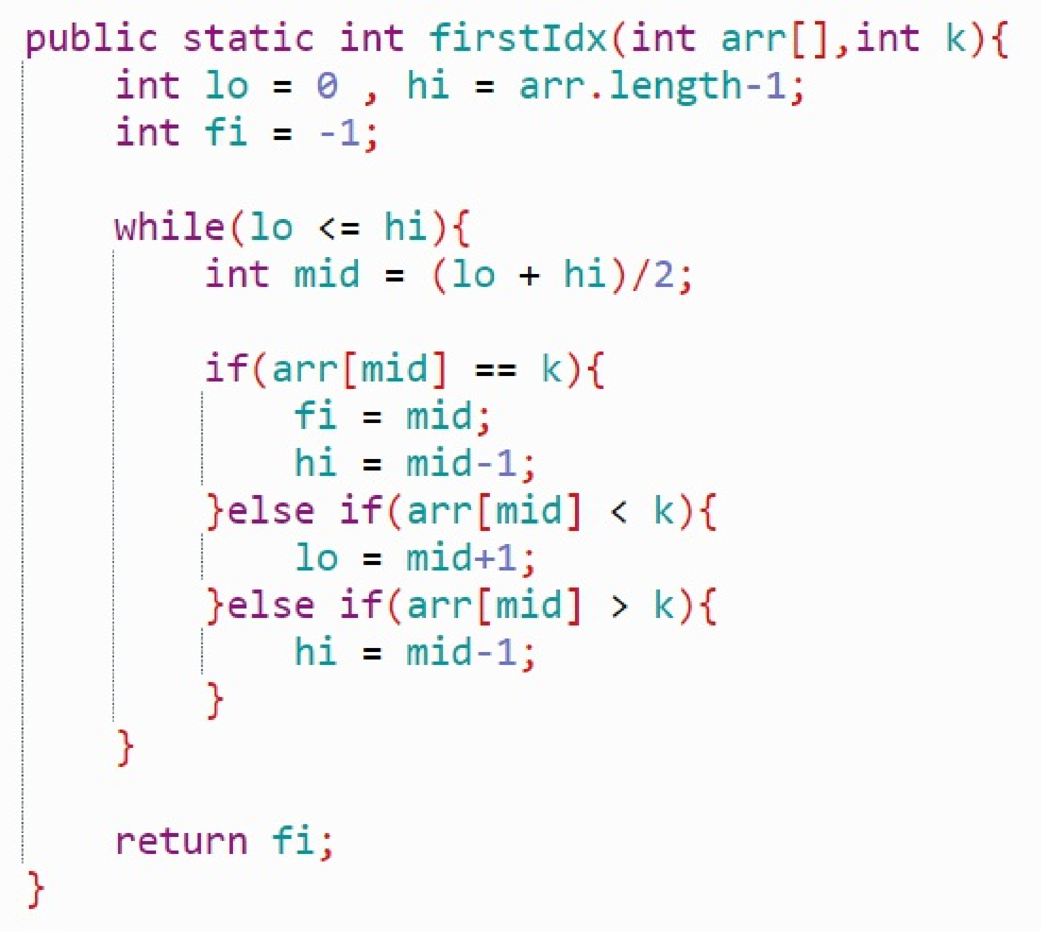
fi = ~~1~~ ~~7~~ 5



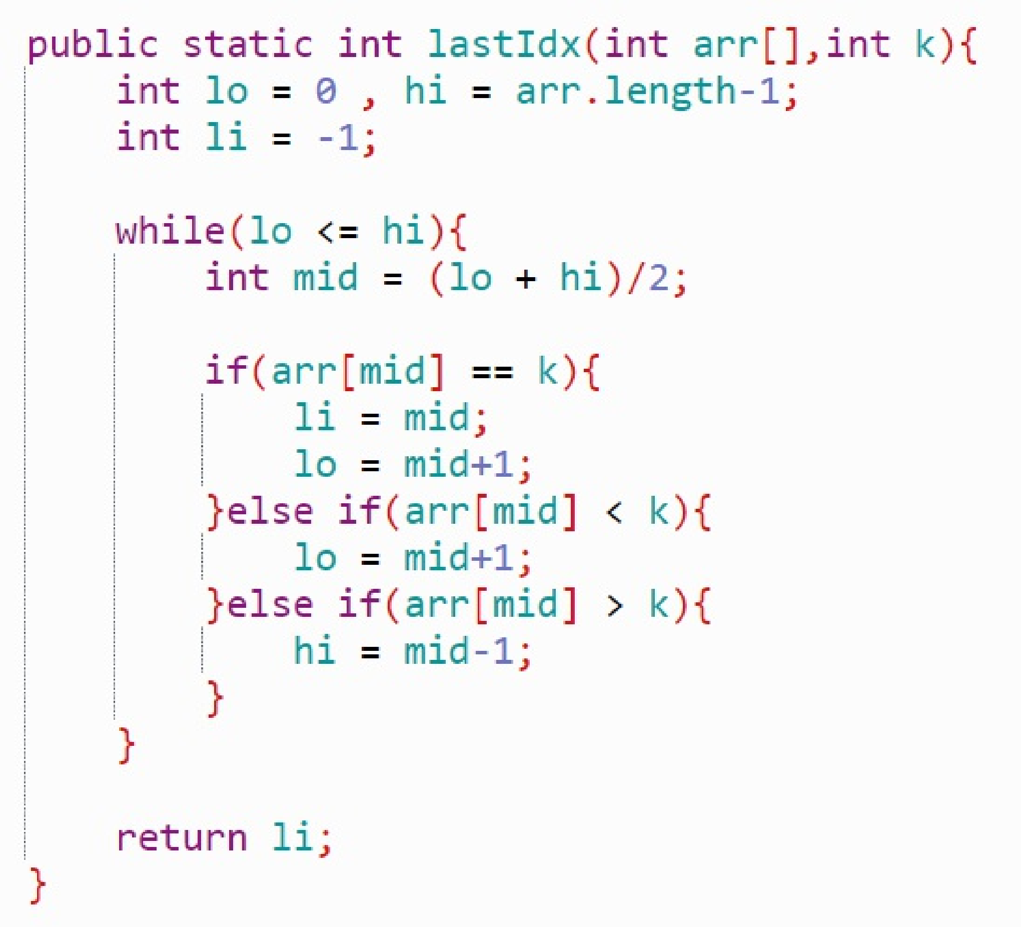
```
int lo = 0 , hi = arr.length-1;
int fi = -1;

while(lo <= hi){
    int mid = (lo + hi)/2;

    if(arr[mid] == k){
        fi = mid;
        hi = mid-1;
    }else if(arr[mid] < k){
        lo = mid+1;
    }else if(arr[mid] > k){
        hi = mid-1;
    }
}

return fi;
```



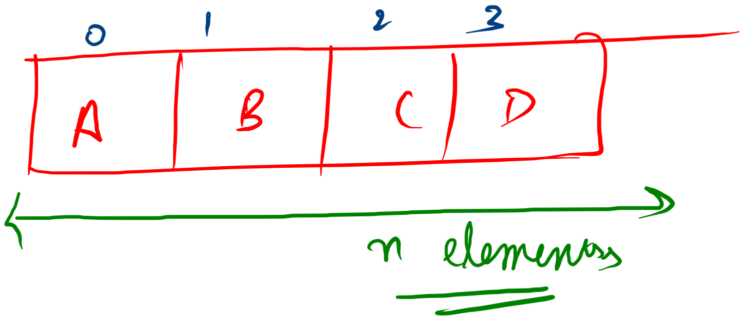
k = 5



```java
public static int firstIdx(int arr[],int k){
    int lo = 0 , hi = arr.length-1;
    int fi = -1;

    while(lo <= hi){
        int mid = (lo + hi)/2;

        if(arr[mid] == k){
            fi = mid;
            hi = mid-1;
        }else if(arr[mid] < k){
            lo = mid+1;
        }else if(arr[mid] > k){
            hi = mid-1;
        }
    }

    return fi;
}
```



```java
public static int lastIdx(int arr[],int k){
    int lo = 0 , hi = arr.length-1;
    int li = -1;

    while(lo <= hi){
        int mid = (lo + hi)/2;

        if(arr[mid] == k){
            li = mid;
            lo = mid+1;
        }else if(arr[mid] < k){
            lo = mid+1;
        }else if(arr[mid] > k){
            hi = mid-1;
        }
    }

    return li;
}
```



Subarrays

↳ Cont. subpart of array

4
A
A B
A B C
A B C D

3
B
B C
B C D

2
C
C D

1
D

$T(4) = 4 + 3 + 2 + 1$

$T(n) = n + (n-1) + (n-2) + \dots\dots + 3 + 2 + 1$

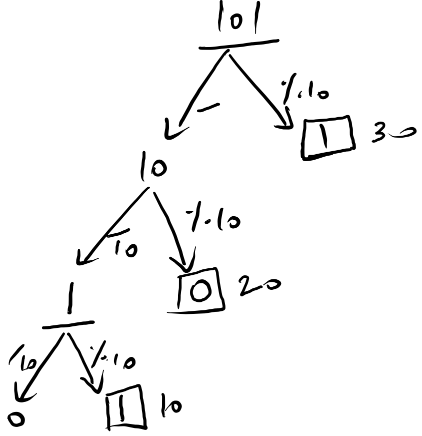
$$\boxed{T(n) = \left[\frac{n(n+1)}{2}\right]}$$ ✓

2^n

A B C D

16

A	B	C	D								
(2)	(2)	(2)	(2)								
0	0	0	0	→	—	—	—	—	→	''''
0	0	0	1	→	—	—	—	D	→	D
0	0	1	0	→	—	—	C	—	→	C
0	0	1	1	→	—	—	C	D	→	CD
0	1	0	0	→	—	B	—	—	→	B
0	1	0	1	→	—	B	—	D	→	BD
0	1	1	0	→	—	B	C	—	→	BC
0	1	1	1	→	—	B	C	D	→	BCD
1	0	0	0	→	A	—	—	—	→	A
1	0	0	1	→	A	—	—	D	→	AD
1	0	1	0	→	A	—	C	—	→	AC
1	0	1	1	→	A	—	C	D	→	ACD
1	1	0	0	→	A	B	—	—	→	AB
1	1	0	1	→	A	B	—	D	→	ABD
1	1	1	0	→	A	B	C	—	→	ABC
1	1	1	1	→	A	B	C	D	→	ABCD



$3 \to 2^3 \to 8$

3
10
20
30

	10 ✓	20 ✓	30 ✓							
0	0	0	0	⇒	-	-	-	-	-	-
1	0	0	1	⇒	-	-	30	-	-	30
2	0	1	0	⇒	-	20	-	-	20	-
3	0	1	1	⇒	-	20	30	-	20	30
4	1	0	0	=	10	-	-	10	-	-
5	1	0	1	=	10	-	30	10	-	30
6	1	1	0	=	10	20	-	10	20	-
7	1	1	1	=	10	20	30	10	20	30

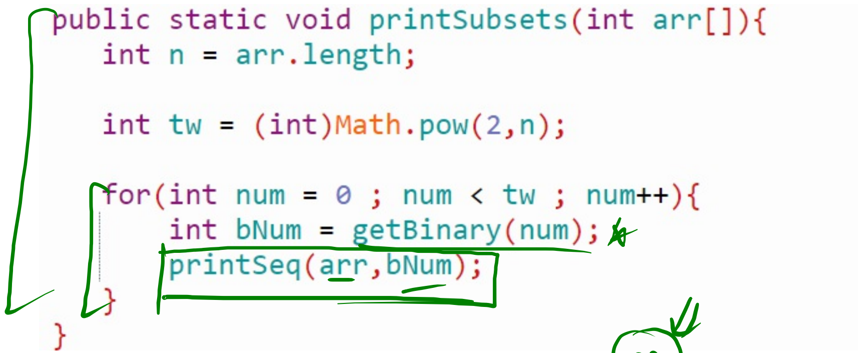
	0	1	2
arr =	10	20	30

String + ___ → String

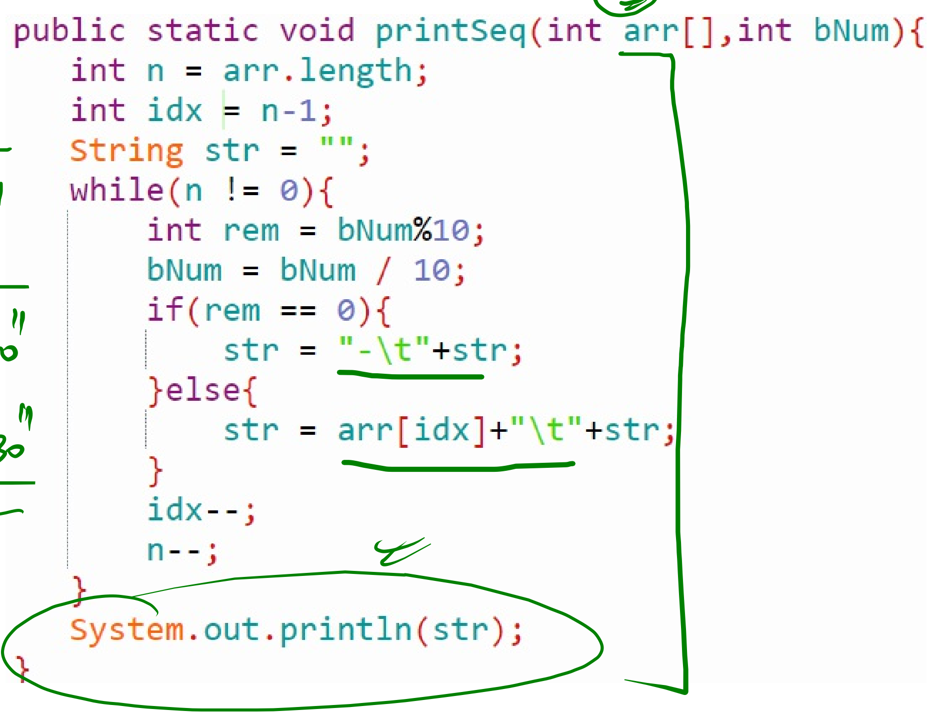
n=3 , idx=2

num	getBinary(num)	print Seq
0	0	- - -
1	1	
2	10	
3	11	
4	100	10 - -
5	101	10 - 30
6	110	
7	111	

n	idx	bNum	rem	str
3	2	101	1	"30"
2	1	10	0	"- 30"
1	0	1	1	"10 - 30"
0	-1	0		

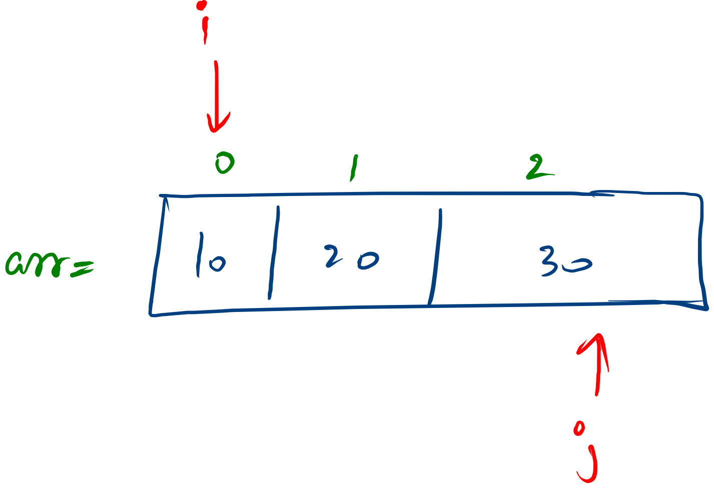


```
public static void printSubsets(int arr[]){
    int n = arr.length;

    int tw = (int)Math.pow(2,n);

    for(int num = 0 ; num < tw ; num++){
        int bNum = getBinary(num);
        printSeq(arr,bNum);
    }
}
```



n

```
public static void printSeq(int arr[],int bNum){
    int n = arr.length;
    int idx = n-1;
    String str = "";
    while(n != 0){
        int rem = bNum%10;
        bNum = bNum / 10;
        if(rem == 0){
            str = "-\t"+str;
        }else{
            str = arr[idx]+"\t"+str;
        }
        idx--;
        n--;
    }
    System.out.println(str);
}
```



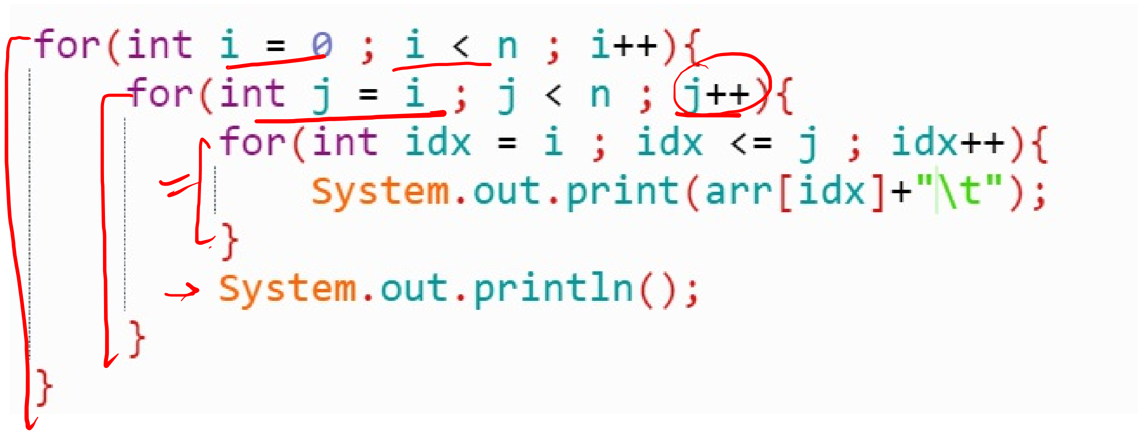
n = 3

n^3

10

→ 10 20

→



```
for(int i = 0 ; i < n ; i++){
    for(int j = i ; j < n ; j++){
        for(int idx = i ; idx <= j ; idx++){
            System.out.print(arr[idx]+"\t");
        }
        System.out.println();
    }
}
```