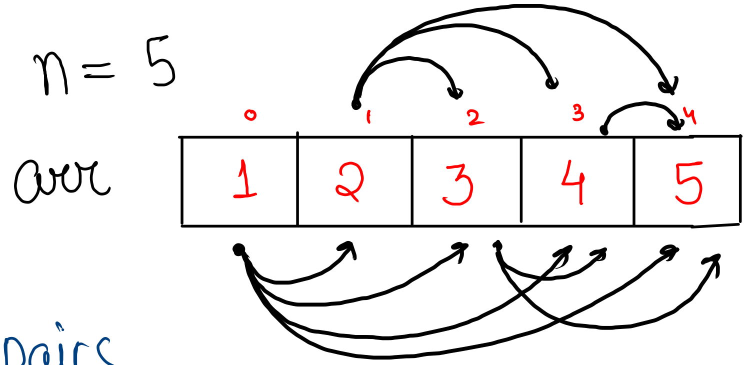
Print Pair



4,1

3,2

pairs

1,2 → (0,1)
1,3 → (0,2)
1,4 → (0,3)
1,5 → (0,4)

2,3 → (1,2)
2,4 → (1,3)
2,5 → (1,4)

3,4 → (2,3)
3,5 → (2,4)

4,5 → (3,4)

Note:- These kind of pairs are called as **Combination**

Note :-

Imp

When we only go in forward direction :- **Combination**

- with repetation
- without repetation

When we only go in Both direction :- **Permutation**

- with repetation
- without repetation

indexes

(i, j)

(0, 1) i = 0, j = 1 → 4

(0, 2)

(0, 3)

(0, 4)

(1, 2) i = 1, j = 2 → 4

(1, 3)

(1, 4)

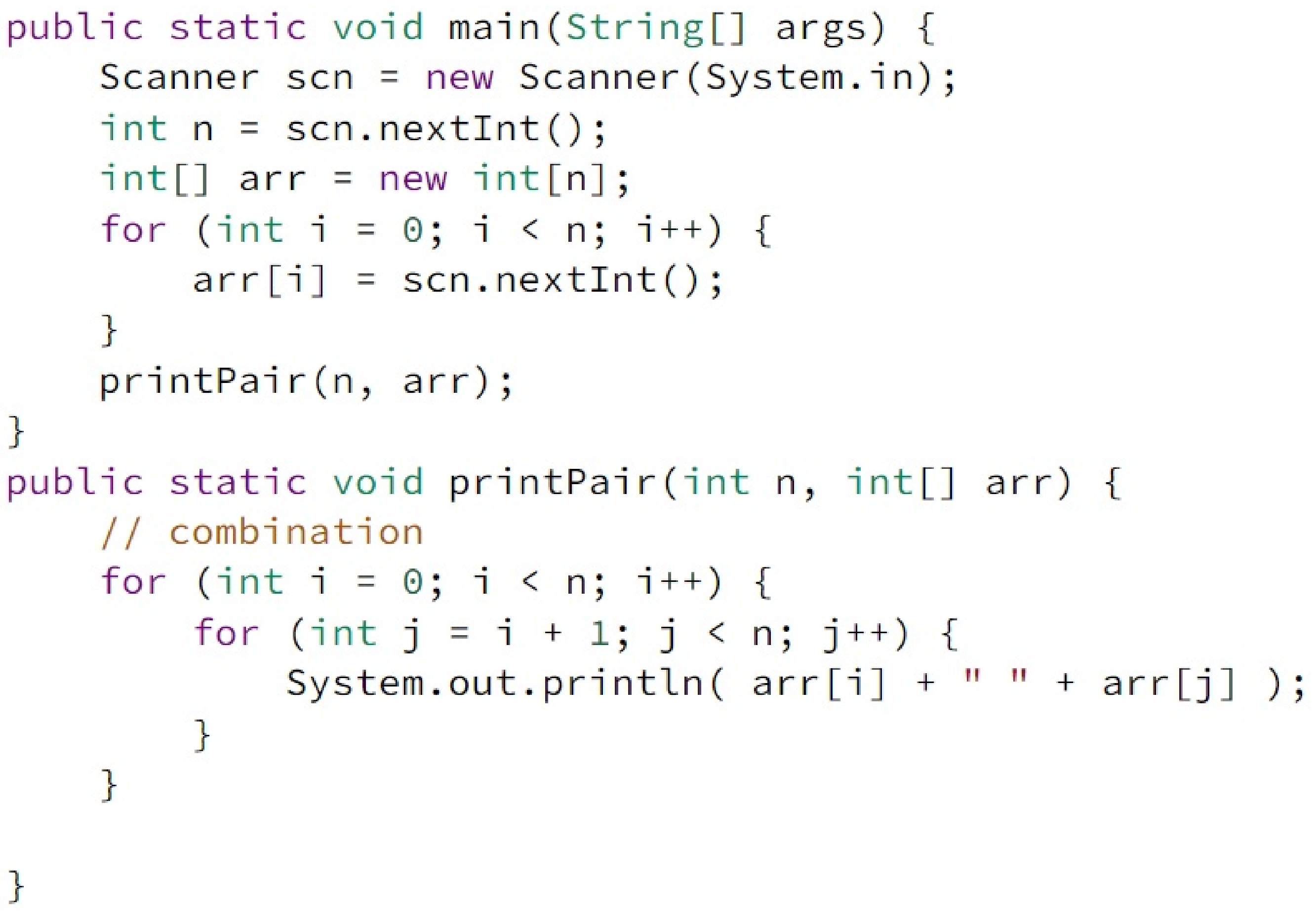
(2, 3) i = 2, j = 3 → 4

(2, 4)

(3, 4) i = 3, j = 4 → 4

for each 'i' value, 'j' is starting from 'i+1' till 'n-1'

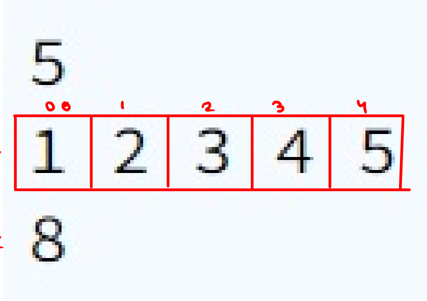
Combination without rep.



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    printPair(n, arr);
}
public static void printPair(int n, int[] arr) {
    // combination
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            System.out.println( arr[i] + " " + arr[j] );
        }
    }

}
```

Find all Combination (target sum)



n = 5

arr =

0	1	2	3	4
1	2	3	4	5

target= 8

pairs

1, 1 = 2

1, 2 = 3

1, 3 = 4

1, 4 = 5

1, 5 = 6

2, 2 = 4

2, 3 = 5

2, 4 = 6

2, 5 = 7

3, 3 = 6

3, 4 = 7

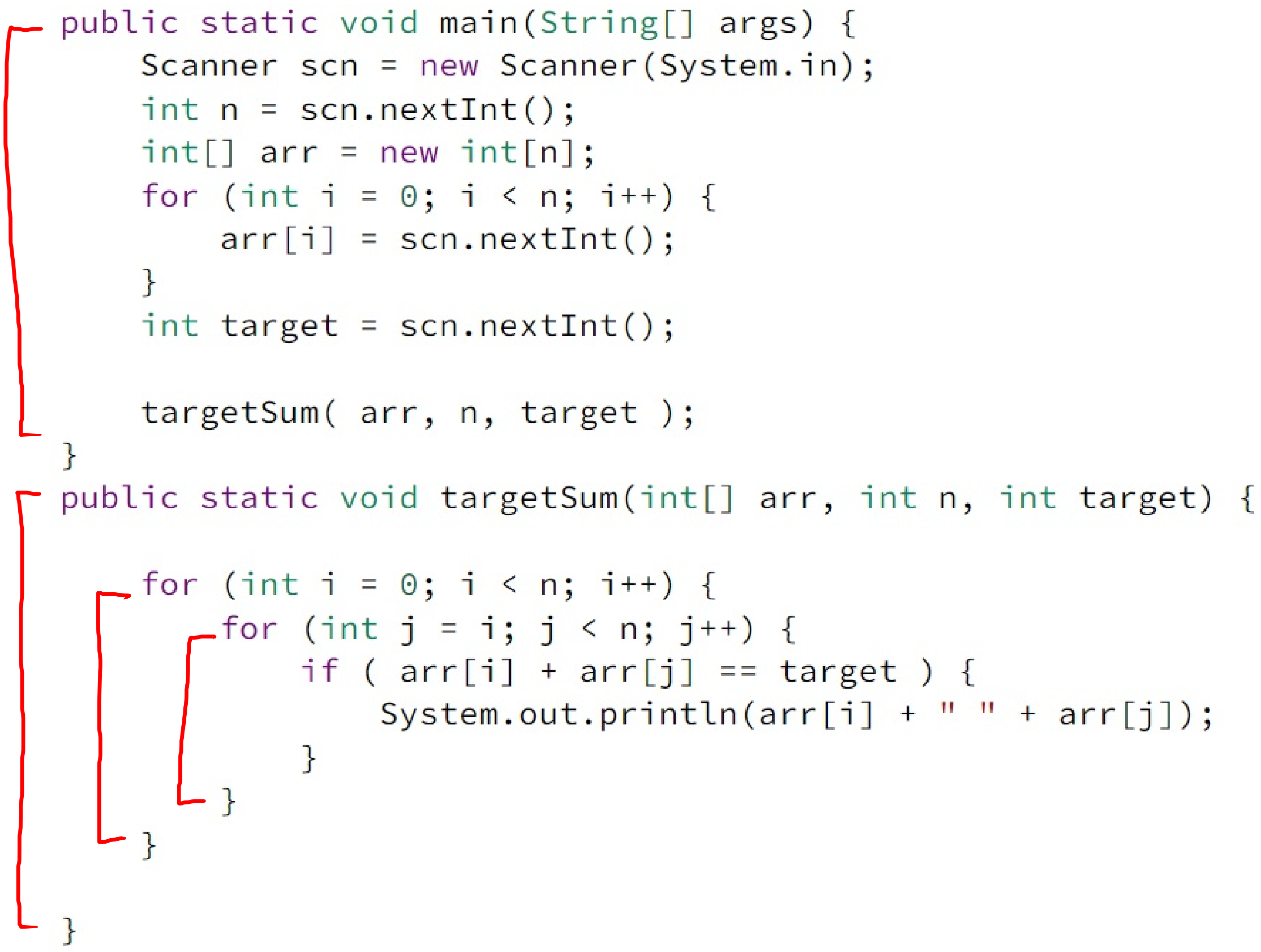
3, 5 = 8 ✓

4, 4 = 8 ✓

4, 5 = 9

5, 5 = 10

// Comb. with rep.



```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    int target = scn.nextInt();

    targetSum( arr, n, target );
}
public static void targetSum(int[] arr, int n, int target) {

    for (int i = 0; i < n; i++) {
        for (int j = i; j < n; j++) {
            if ( arr[i] + arr[j] == target ) {
                System.out.println(arr[i] + " " + arr[j]);
            }
        }
    }

}
```

Greater Than Me

(P without R)

n = 7

arr =

0	1	2	3	4	5	6
5	-2	3	1	7	4	3

counting of strickly greater

ans

i = 0,   1

i = 1,   6

i = 2,   3

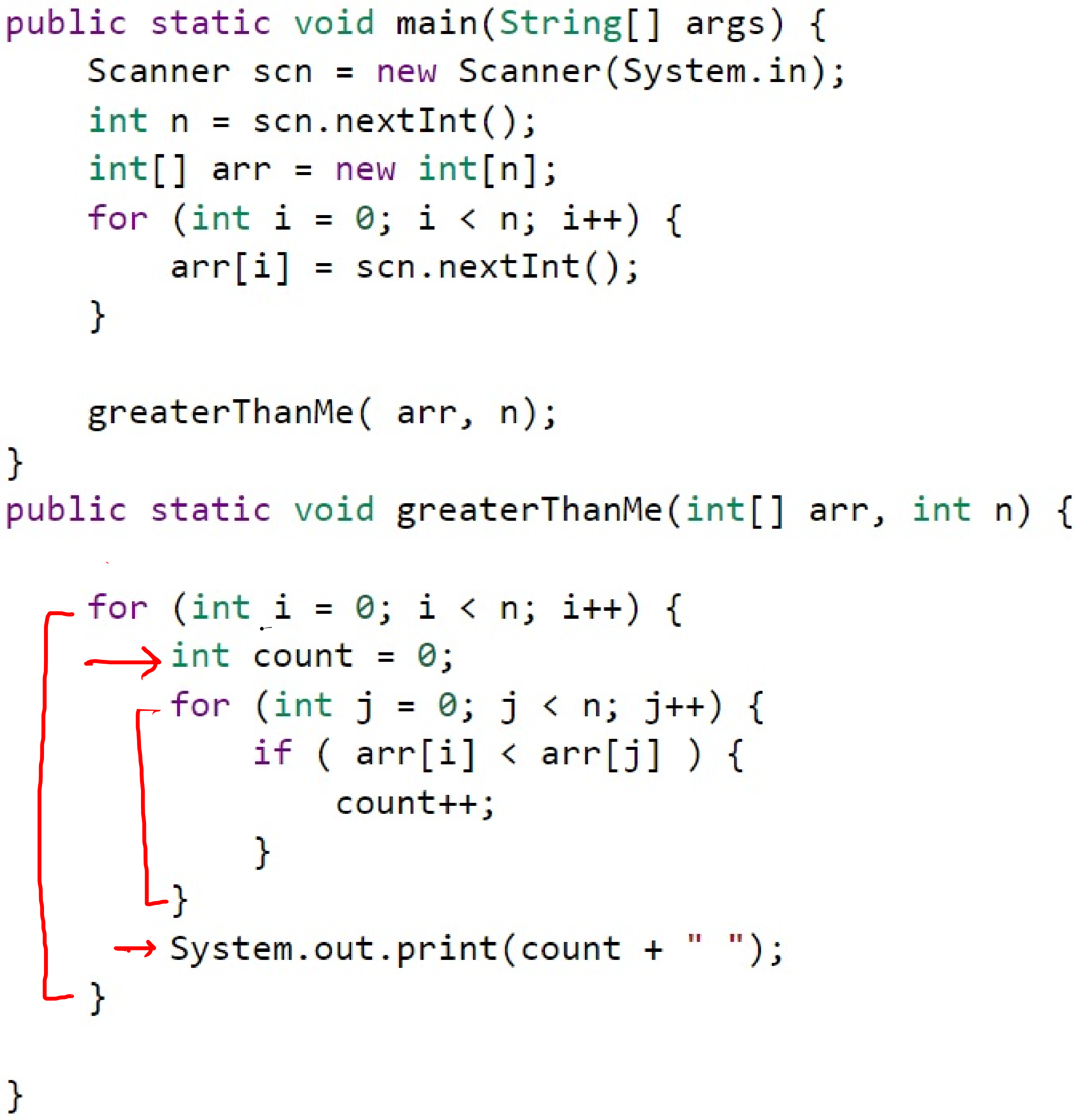
i = 3,   5

i = 4,   0

i = 5,   2

i = 6,   3

code



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    greaterThanMe( arr, n);
}
public static void greaterThanMe(int[] arr, int n) {

    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = 0; j < n; j++) {
            if ( arr[i] < arr[j] ) {
                count++;
            }
        }
        System.out.print(count + " ");
    }

}
```

	0	1	2	3
arr	2	3	-2	4

i= 0, count= ~~0~~ ~~1~~ 2

j=0, (2<2) ✗
j=1, (2<3) ✓
j=2, (2<-2) ✗
j=3, (2<4) ✓

i= 1, count= ~~0~~ 1

j= 0, (3<2) ✗
j=1, (3<3) ✗
j=2, (3< -2) ✗
j= 3, (3< 4) ✓

i= 2, count= ~~0~~ ~~1~~ ~~2~~ 3

j= 0, (-2 < 2) ✓
j= 1, (-2 < 3) ✓
j= 2, (-2< -2) ✗
j= 3, (-2 < 4) ✓

i= 3, count= 0

j= 0, (4 < 2) ✗
j= 1, (4 < 3) ✗
j= 2, (4 <-2) ✗
j= 3, (4 < 4) ✗

Greater At Right

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    greaterThanMe( arr, n);
}
public static void greaterThanMe(int[] arr, int n) {

    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = i + 1; j < n; j++) {
            if ( arr[i] < arr[j] ) {
                count++;
            }
        }
        System.out.print(count + " ");
    }

}
```