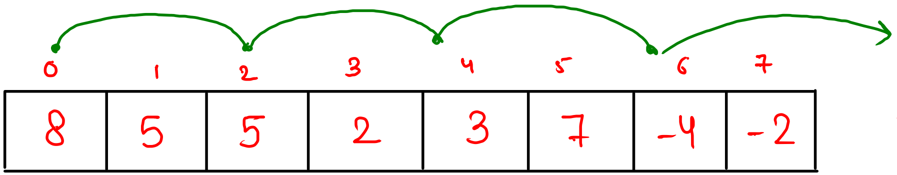
Note :-

```
boolean[] arr = new boolean[5];
```

0	1	2	3	4
false	false	false	false	false

Print Alternate Array Elements Linewise



arr =

0	1	2	3	4	5	6	7
8	5	5	2	3	7	-4	-2

size = 8

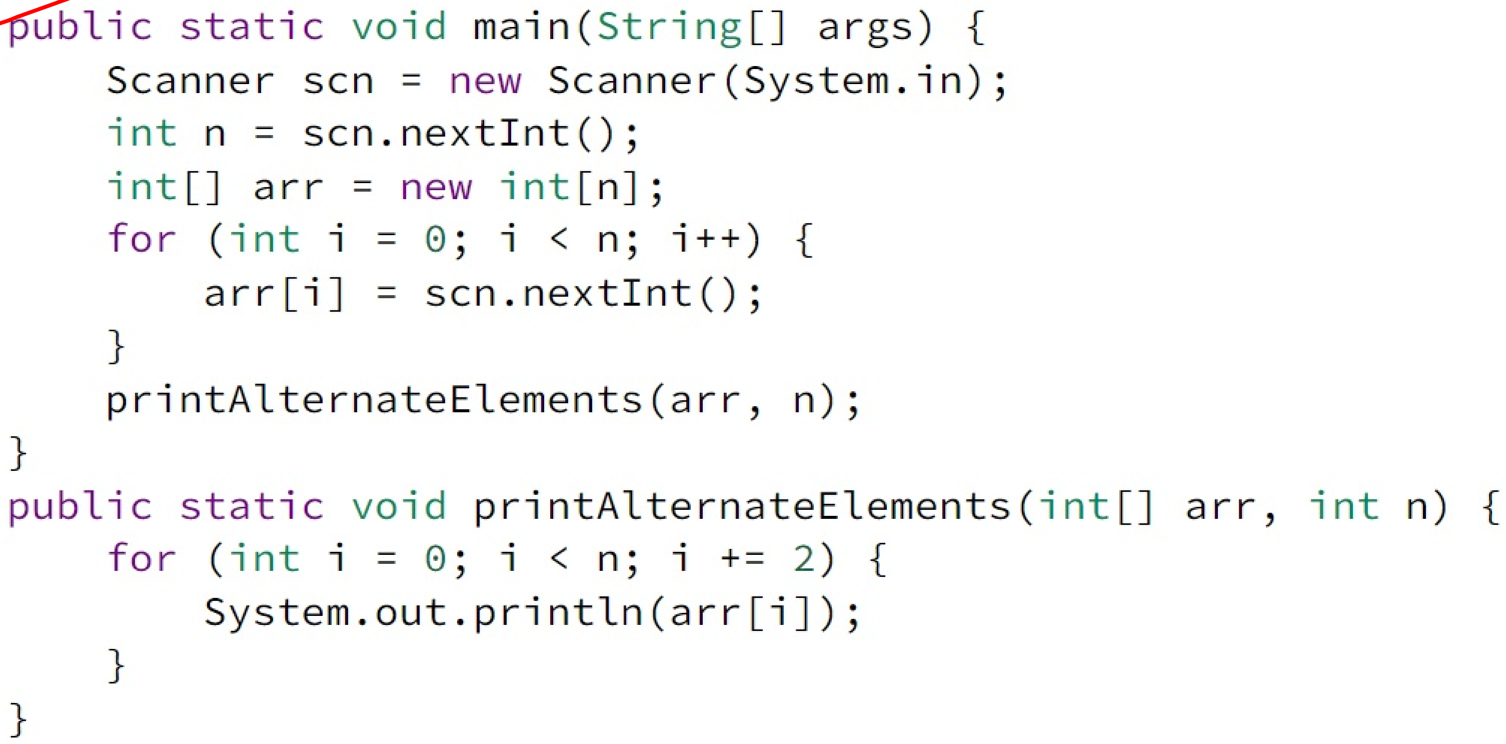
ans :- 8 5 3 -4

one liner :- from 0 to n by +2

code)

```
for( int i=0; i<n; i+=2){
        Syso( arr[i]+"  ");
}
```

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();
    }
    printAlternateElements(arr, n);
}
public static void printAlternateElements(int[] arr, int n) {
    for (int i = 0; i < n; i += 2) {
        System.out.println(arr[i]);
    }
}
```

i → index

arr[i] → value at ith index

Print Array Elements Reverse linewise

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

    printReverse(arr, n);
}
public static void printReverse(int[] arr, int n) {
    for (int i = n - 1; i >= 0; i--) {
        System.out.print(arr[i] + " ");
    }
}
```

Print Array element if index divisible by 3

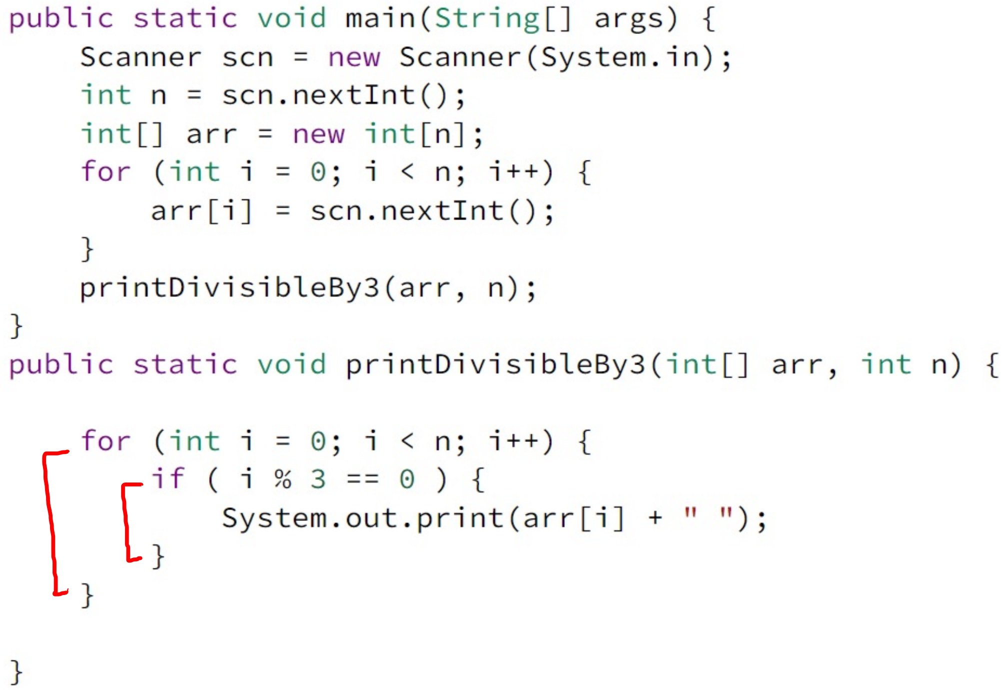
print elements when index is divisible by 3

	0 ✓	1 ✗	2 ✗	3 ✓	4 ✗	5 ✗	6 ✓	7 ✗
arr =	8	5	5	2	3	7	-4	-2

o/p :- 8 2 -4

code

i % 3 == 0   true



```
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();
    }
    printDivisibleBy3(arr, n);
}
public static void printDivisibleBy3(int[] arr, int n) {

    for (int i = 0; i < n; i++) {
        if ( i % 3 == 0 ) {
            System.out.print(arr[i] + " ");
        }
    }

}
```

Check if two arrays are identical?

if arrays are equal

- → size of both the arrays should be same
- → every element at the coorosponding index should be same

Ex :-

$n = 4$

arr1 =

0	1	2	3
5	-2	4	3

$m = 4$

arr2 =

0	1	2	3
5	-2	5	3

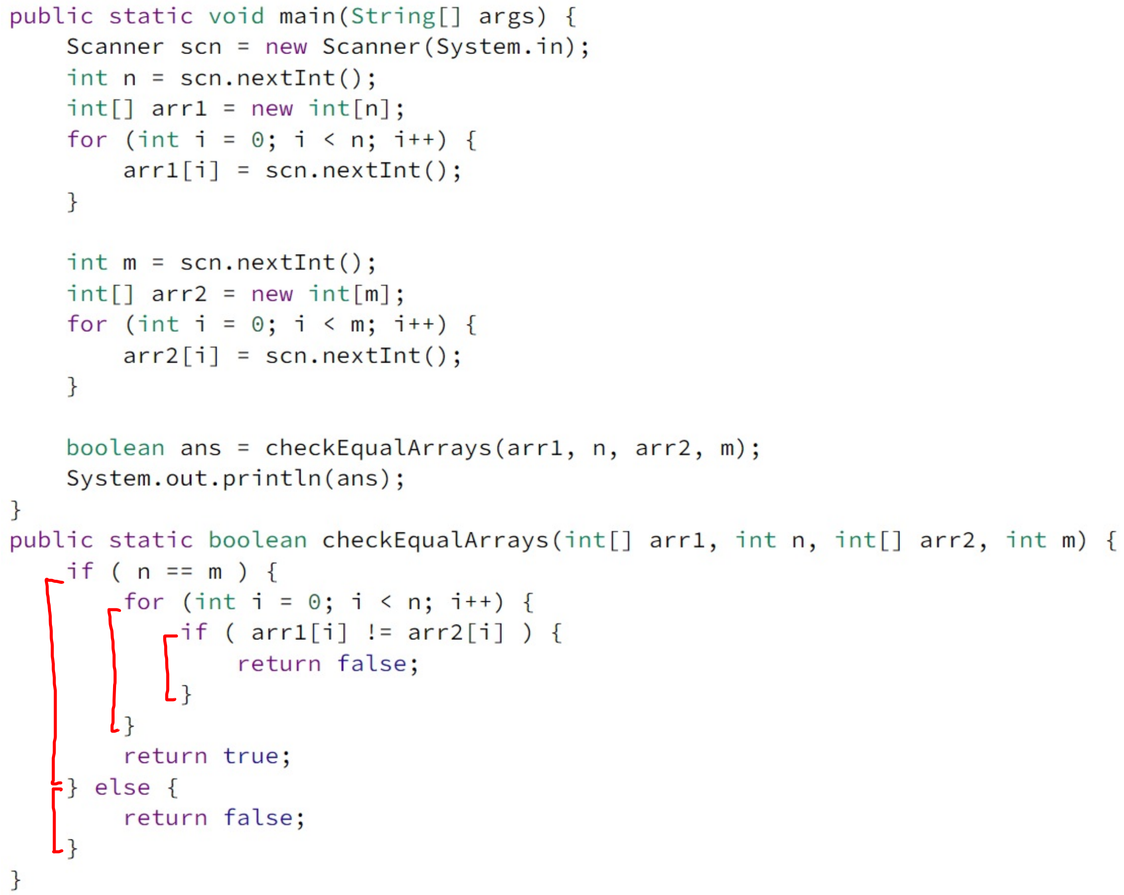
ans = false

Note :- we always check opposite condition, from which is asked in the question.

psudo code

```
1) check if m == n
    1.1) loop from 0 to n
        1.1.1) check if arr1[i] != arr2[i]
               return false
    1.2) return true
2) else
       return false
```

code



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr1 = new int[n];
    for (int i = 0; i < n; i++) {
        arr1[i] = scn.nextInt();
    }

    int m = scn.nextInt();
    int[] arr2 = new int[m];
    for (int i = 0; i < m; i++) {
        arr2[i] = scn.nextInt();
    }

    boolean ans = checkEqualArrays(arr1, n, arr2, m);
    System.out.println(ans);
}
public static boolean checkEqualArrays(int[] arr1, int n, int[] arr2, int m) {
    if ( n == m ) {
        for (int i = 0; i < n; i++) {
            if ( arr1[i] != arr2[i] ) {
                return false;
            }
        }
        return true;
    } else {
        return false;
    }
}
```

n=6

	0	1	2	3	4	5
arr1 =	5	2	3	-4	2	3

m=6

	0	1	2	3	4	5
arr2 =	5	2	3	-4	2	3

i = 0, (5 != 5) false

i = 1, (2 != 2) false

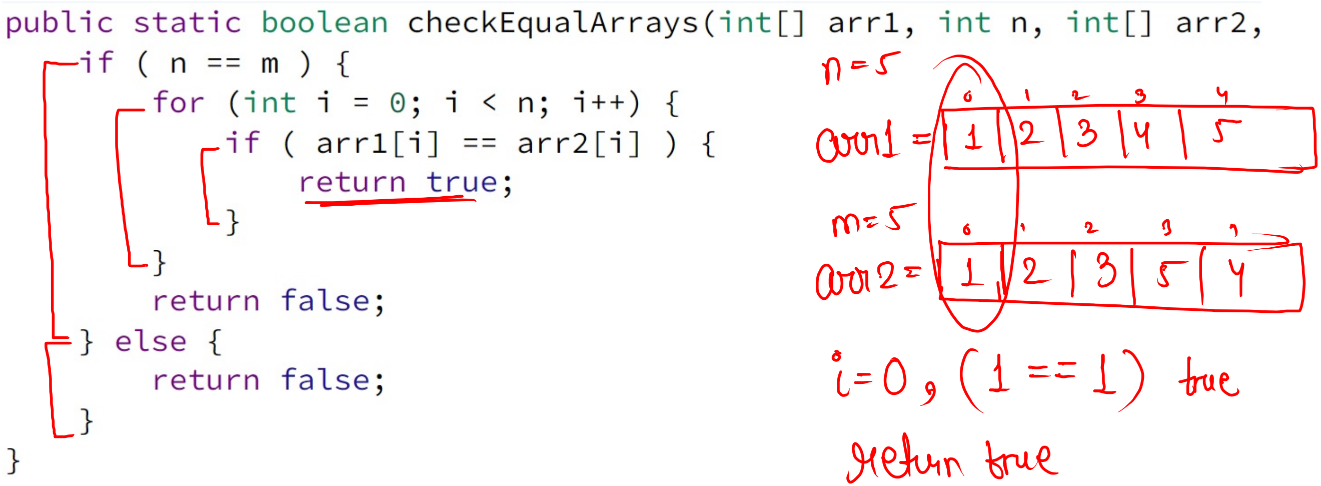
i = 2, (3 != 3) false

i = 3, (-4 != -4) false

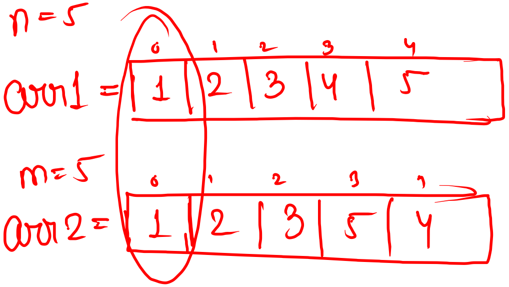
i = 4, (2 != 2) false

i = 5, (3 != 3) false

Code



```
public static boolean checkEqualArrays(int[] arr1, int n, int[] arr2,
    if ( n == m ) {
        for (int i = 0; i < n; i++) {
            if ( arr1[i] == arr2[i] ) {
                return true;
            }
        }
        return false;
    } else {
        return false;
    }
}
```



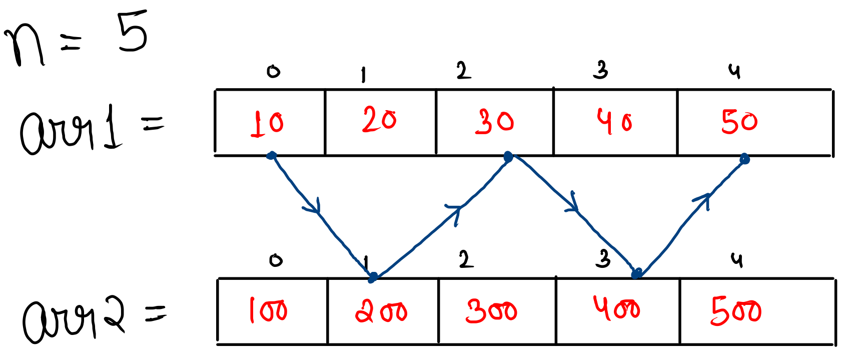
i=0, (1 == 1) true

return true

This code is wrong

Print two arrays alternately

n = 5



arr1 =

0	1	2	3	4
10	20	30	40	50

arr2 =

0	1	2	3	4
100	200	300	400	500

index	0	1	2	3	4
	↓	↓	↓	↓	↓
	arr1	arr2	arr1	arr2	arr1

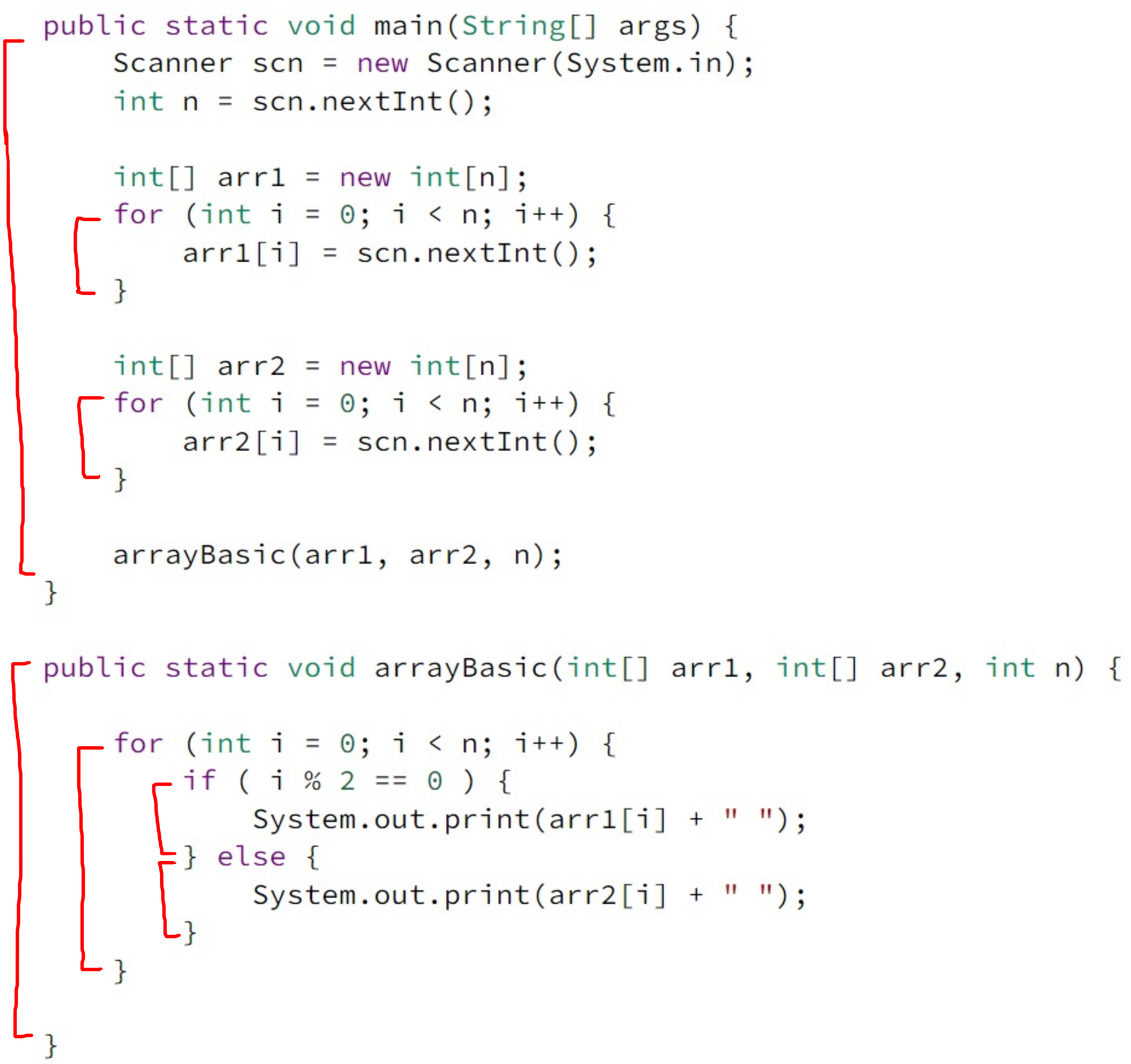
even index :- print arr1

odd index :- print arr2

psudo code

```
1) loop from 0 to n
    1.1) check i%2==0
            Syso ( arr1[i] );
    else
            Syso ( arr2[i] );
```

code

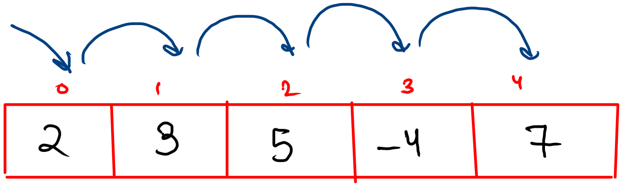


```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    int[] arr1 = new int[n];
    for (int i = 0; i < n; i++) {
        arr1[i] = scn.nextInt();
    }

    int[] arr2 = new int[n];
    for (int i = 0; i < n; i++) {
        arr2[i] = scn.nextInt();
    }

    arrayBasic(arr1, arr2, n);
}

public static void arrayBasic(int[] arr1, int[] arr2, int n) {

    for (int i = 0; i < n; i++) {
        if ( i % 2 == 0 ) {
            System.out.print(arr1[i] + " ");
        } else {
            System.out.print(arr2[i] + " ");
        }
    }

}
```

→ Arrays Searching

Check if x is present in array or not

n = 5



arr =

0	1	2	3	4
2	3	5	-4	7

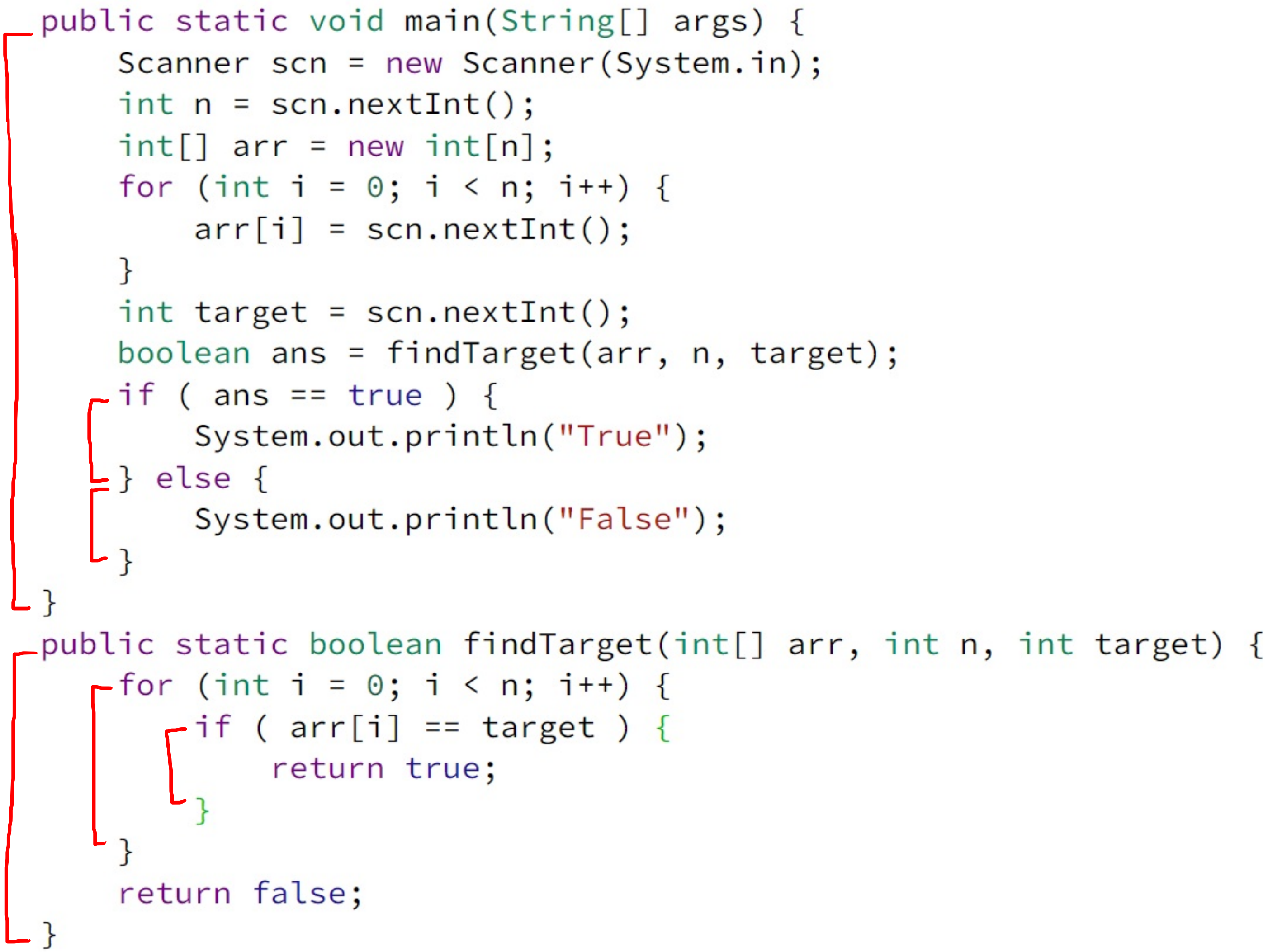
target = -4

linear Search

approach

↳ [we are going to check each and every element from start to end, and if at any time our current element got equal to target then return true]

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();
     }
     int target = scn.nextInt();
     boolean ans = findTarget(arr, n, target);
     if ( ans == true ) {
          System.out.println("True");
     } else {
          System.out.println("False");
     }
}
public static boolean findTarget(int[] arr, int n, int target) {
     for (int i = 0; i < n; i++) {
          if ( arr[i] == target ) {
               return true;
          }
     }
     return false;
}
```