⇒ Subarray / Substring

(sub - part of array)

properties :-

- ★ → subarray is always continuous
  → subarray always built in forward direction.

arr =

0	1	2	3
1	5	3	2

(i, j)

Subarrays

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

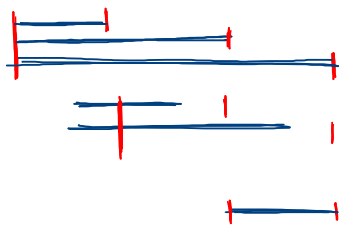
n=4

arr =

0	1	2	3
1	5	3	2

i = starting index = 0

j = ending index = 0, 1, 2, 3



i = 1
j = 1, 2, 3

i = 2
j = 2, 3

i = 3, j = 3

```
for(int i=0; i<n; i++) {
    for(int j=i; j<n; j++) {
        print from i to j
    }
}
```

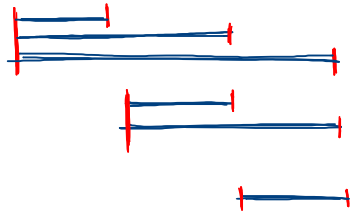
n = 4

arr =

0	1	2	3
1	5	3	2

i = starting index = 0

j = ending index = 0, 1, 2, 3



i = 1
j = 1, 2, 3

i = 2
j = 2, 3

i = 3, j = 3

```
for(int i=0; i<n; i++) {
    for(int j=i; j<n; j++) {
        print from i to j
    }
}
```

Code (print all subarrays)

```
public static void main(String[] args) {
    int n = 4;
    int[] arr = {1, 5, 3, 2};

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

Print All Substrings

T.C = $O(n^3)$

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String str = scn.nextLine();

    int n = str.length();
    for (int i = 0; i < n; i++) {
        for (int j = i; j < n; j++) {
            print(str, i, j);
        }
    }
}
public static void print(String str, int i, int j) {
    for (int k = i; k <= j; k++) {
        System.out.print(str.charAt(k));
    }
    System.out.println();
}
```

template

$n^2 * n$

Sum Equals Zero

arr =

0	1	2	3	4
5	-2	3	-1	4

Subarrays

(5) 5
(3) 5 -2
(6) 5 -2 3
(5) 5 -2 3 -1
(9) 5 -2 3 -1 4
(-2) -2
(1) -2 3
(0) -2 3 -1
-2 3 -1 4
3
3 -1
3 -1 4
-1
-1 4
4

return true

psudo code

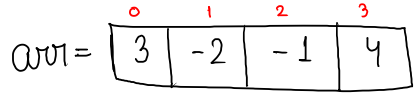
1) loop from 0 to n
2) loop from i to n
3) find sum from i to j
4) if sum == 0
return true;

return false;

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();
    }

    System.out.println(sumEqualsZero(arr, n));
}
public static boolean sumEqualsZero(int[] arr, int n) {
    for (int i = 0; i < n; i++) {
        for (int j = i; j < n; j++) {
            int sum = checkSum(arr, i, j);
            if (sum == 0) {
                return true;
            }
        }
    }
    return false;
}
public static int checkSum(int[] arr, int x, int y) {
    int sum = 0;
    for (int i = x; i <= y; i++) {
        sum += arr[i];
    }
    return sum;
}
```



sum = ~~3~~ ~~4~~ ~~05~~

3
3 -2
3 -2 -1
3 -2 -1 4
-2
-2 -1
-2 -1 4
-1
-1 4
4

Max Subarray 2

↳ n = 5

arr =

0	1	2	3	4
-1	2	3	-2	1

→ First Approach

psudo code

1) loop from 0 to n

1.1) loop from i to n

1.1.1) find sum

ans = max(ans, sum);

T.C = $O(n^3)$

⇒ maximum sum subarray in linear time

↳ Kadane's Algorithm $O(n)$

arr =

0	1	2	3
3	-20	4	7

i

maxSum = ~~-∞~~ ~~3~~ ~~4~~ 11

sum_so_far = ~~0~~ ~~3~~ ~~-17~~ ~~4~~ 11

```
if (sum_so_far < 0) {
    sum_so_far = arr[i];
} else {
    sum_so_far += arr[i]
}

if (sum_so_far > maxSum) {
    maxSum = sum_so_far;
}
```

```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();
    }

    System.out.println(maxSumSubarray(arr, n));
}
public static int maxSumSubarray(int[] arr, int n) {
    int sumSoFar = 0;
    int maxSum = Integer.MIN_VALUE;
    for (int i = 0; i < n; i++) {

        if ( sumSoFar < 0 ) {
            sumSoFar = arr[i];
        } else {
            sumSoFar += arr[i];
        }

        if ( sumSoFar > maxSum ) {
            maxSum = sumSoFar;
        }
    }
    return maxSum;
}
```