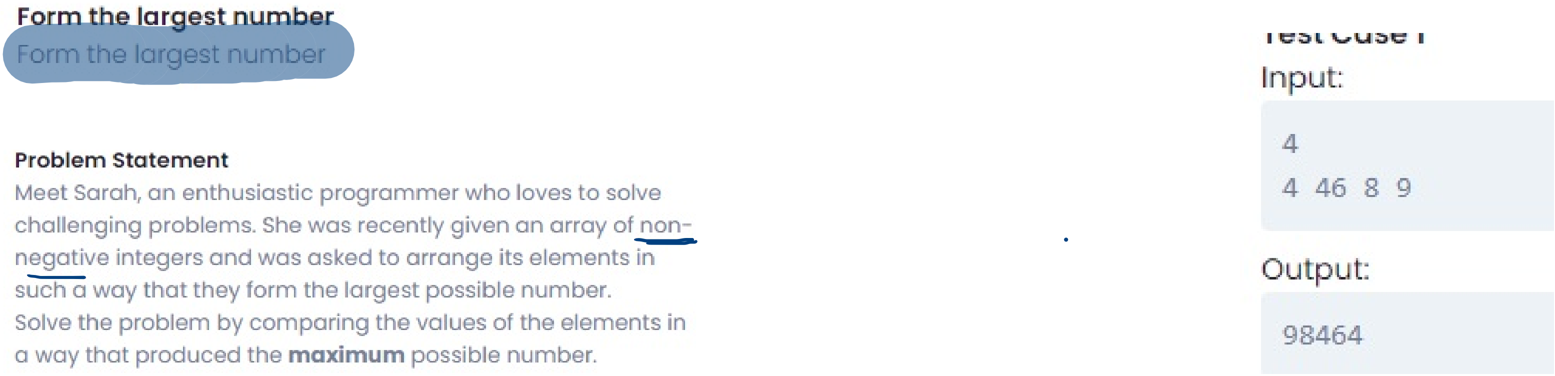


Form the largest number

Form the largest number

Problem Statement

Meet Sarah, an enthusiastic programmer who loves to solve challenging problems. She was recently given an array of non-negative integers and was asked to arrange its elements in such a way that they form the largest possible number.
Solve the problem by comparing the values of the elements in a way that produced the **maximum** possible number.

Test Case 1

Input:

```
4
4  46  8  9
```

Output:

```
98464
```

4	46	8	9
0	1	2	3

greatest.

84649 > 49,846

98464 ✓ → greatest.

1	2	4	3

→ 4321

all single digit

↳ sort desc.

eg.

a.

3	30

→ ? ✗

✗ 303, 330 ✓

b.

3	39

✓ → ?

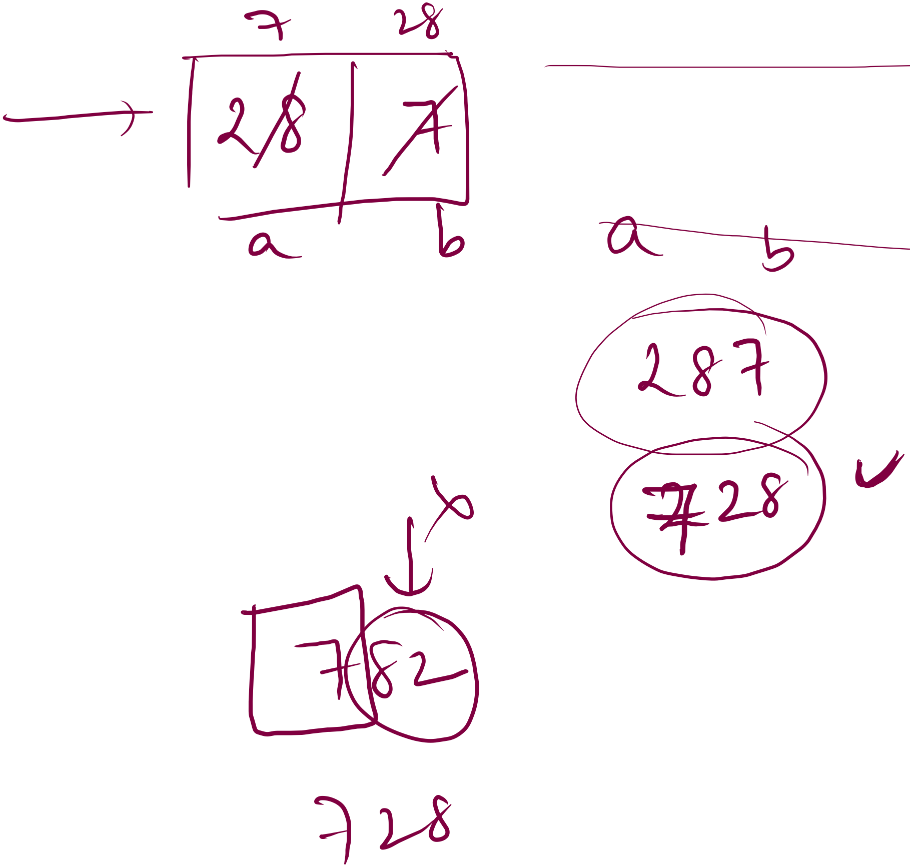
✗ 339 | 393 ✓

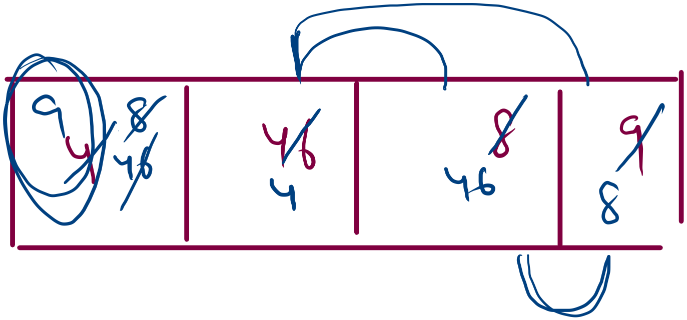
3	30

✓ ✓

330 ✓

303


7
28
28
7
a
b
a
b
287
728
✓
x
782
728



t - s

s → "4 46"

t → "464"

s → 46 8

t → 846

s → 8 9

t → 98

a b

4 46

4+46 = "446" → s

46+4 = "464" → t

b - a.

a → "446"

b → "464"

b. compareTo(a).

t.compareTo(s);

int

t - s

s - t

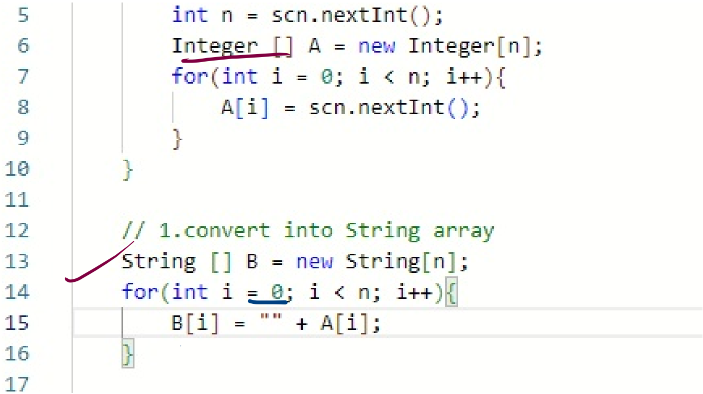
s.compareTo(t);

int 4 $\Rightarrow$ $4 + 46$? $= 50$

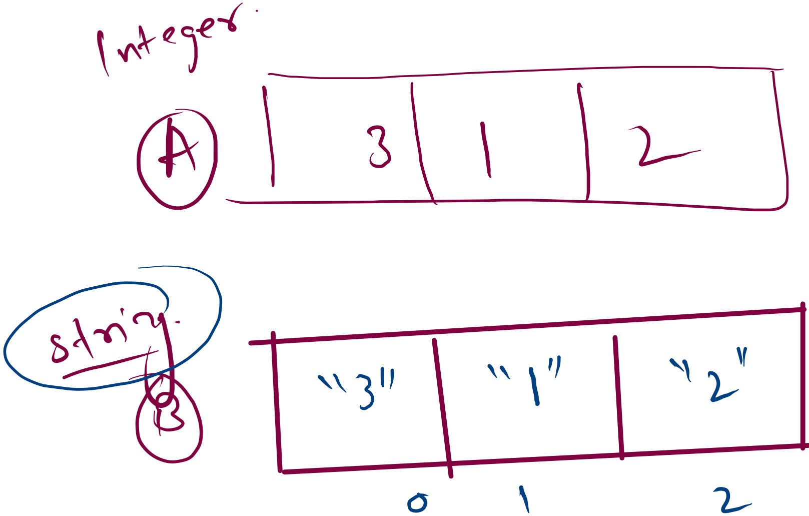
int 46

string 4 $\Rightarrow$ $4 + 46 = \boxed{446}$

string 46

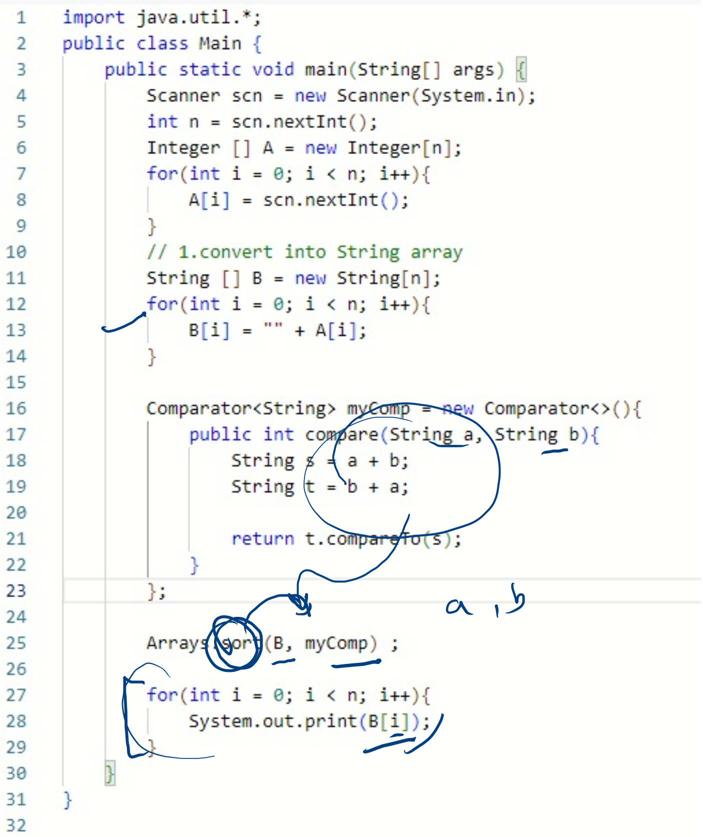


```
        int n = scn.nextInt();
        Integer [] A = new Integer[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
    }

    // 1.convert into String array
    String [] B = new String[n];
    for(int i = 0; i < n; i++){
        B[i] = "" + A[i];
    }
```

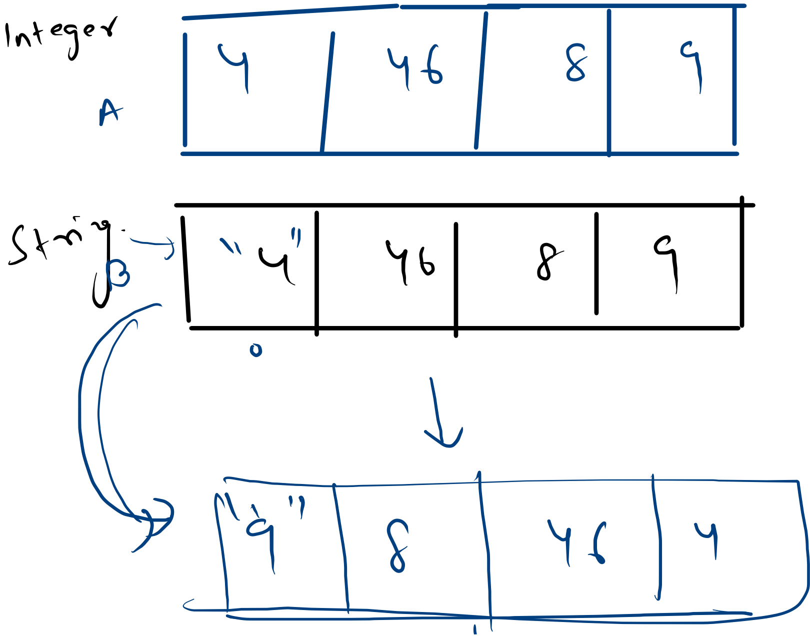


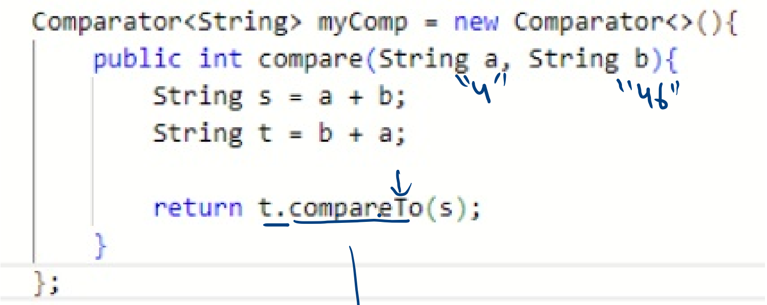
"" + A[i]

"" + 3 ⇒ "3"



```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        Integer [] A = new Integer[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        // 1.convert into String array
        String [] B = new String[n];
        for(int i = 0; i < n; i++){
            B[i] = "" + A[i];
        }

        Comparator<String> myComp = new Comparator<>(){
            public int compare(String a, String b){
                String s = a + b;
                String t = b + a;

                return t.compareTo(s);
            }
        };

        Arrays.sort(B, myComp) ;

        for(int i = 0; i < n; i++){
            System.out.print(B[i]);
        }
    }
}
```



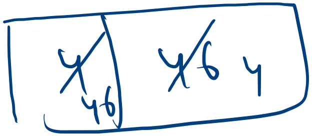


```
Comparator<String> myComp = new Comparator<>(){
    public int compare(String a, String b){
        String s = a + b;
        String t = b + a;

        return t.compareTo(s);
    }
};
```

"4" "46"

int — 0
— +ve t>s
— -ve t<s

t = s

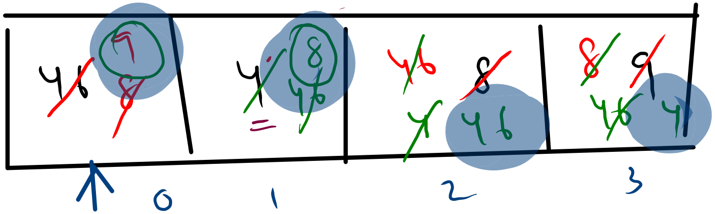


t - s

s → "446"
t → "464" ✓

464 > 446.

9 8 46 4



2 3

✓

1 2

1 3

a b

4 46

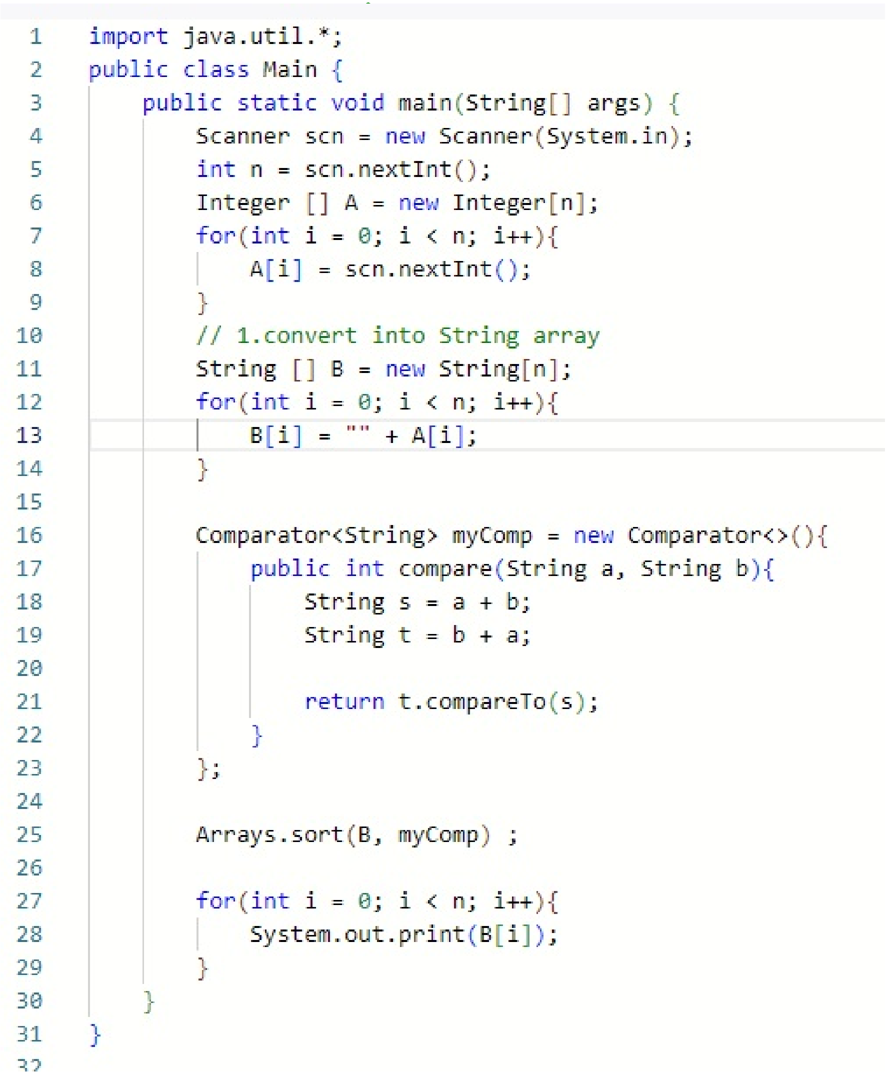
8→ 4 46

t→ 464

t.compareTo(s)

t < s -ve

t > s +ve

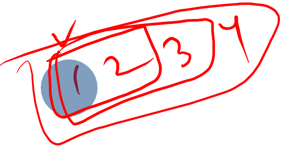


```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        Integer [] A = new Integer[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        // 1.convert into String array
        String [] B = new String[n];
        for(int i = 0; i < n; i++){
            B[i] = "" + A[i];
        }

        Comparator<String> myComp = new Comparator<>(){
            public int compare(String a, String b){
                String s = a + b;
                String t = b + a;

                return t.compareTo(s);
            }
        };

        Arrays.sort(B, myComp) ;

        for(int i = 0; i < n; i++){
            System.out.print(B[i]);
        }
    }
}
```

Subarrays:

1	2	3	4
0	1	2	3

?



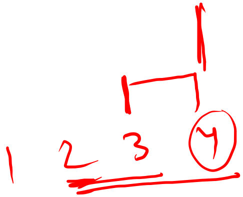
1
1 2
1 2 3
1 2 3 4

2
2 3
2 3 4

3
3 4

4

→ subarrays



5	7	3	2
0	1	2	3

$n=4.$

$1+2+3+4$

$=10$

4

5

5 7

5 7 3

5 7 3 2

3

7

7 3

7 3 2

2

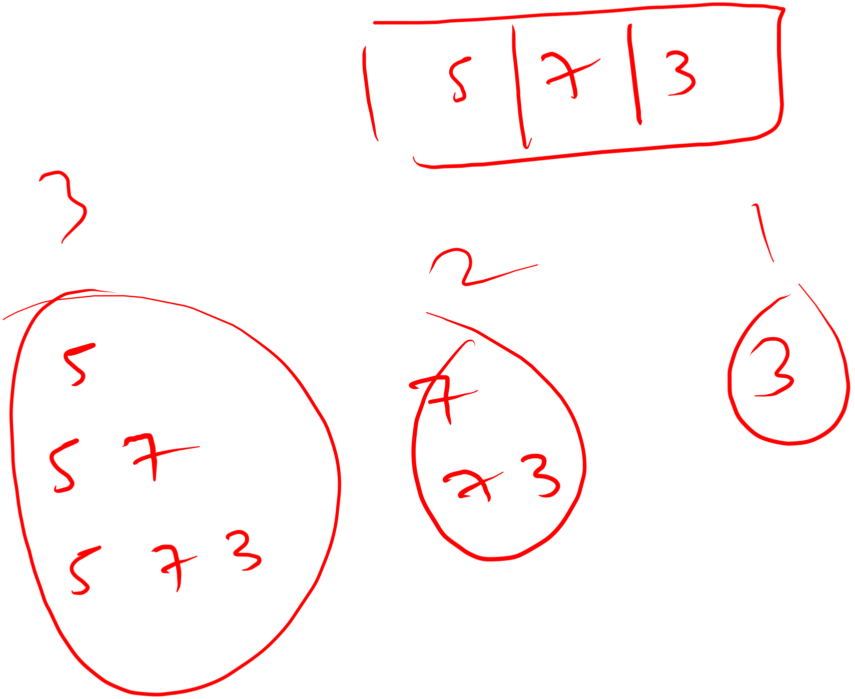
3

3 2

1

2

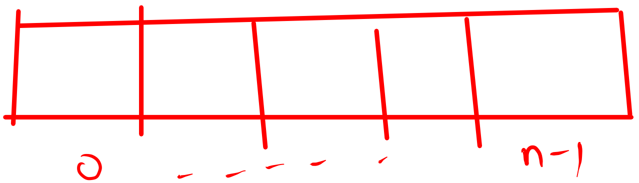
$n = 3$



$1 + 2 + 3 = 6$

$n = 5$?

<u>count ?</u>



$n = \textcircled{n}$

?

$1 + 2 + 3 + 4 + \dots + n$?

$$\frac{n(n+1)}{2} = \text{sum of } n.$$

5	7	3	2
0 ✓	1	2 ✓	3

5

5 7

5 7 3

5 7 3 2

7

7 3

7 3 2

3

3 2

2

start = 0	end 0 1 2 3
1	1 2 3
2	2 3
3	3

1<4

```
for ( start = 0 ; st < n ; st++)
    for ( end = st ; end < n ; end++)
    {
    }
}
```

st | end → print range.

start = 0 ✓

0<4 ✓

0<n

end = 0 1 2 3

1<4 ✓

Start = 1

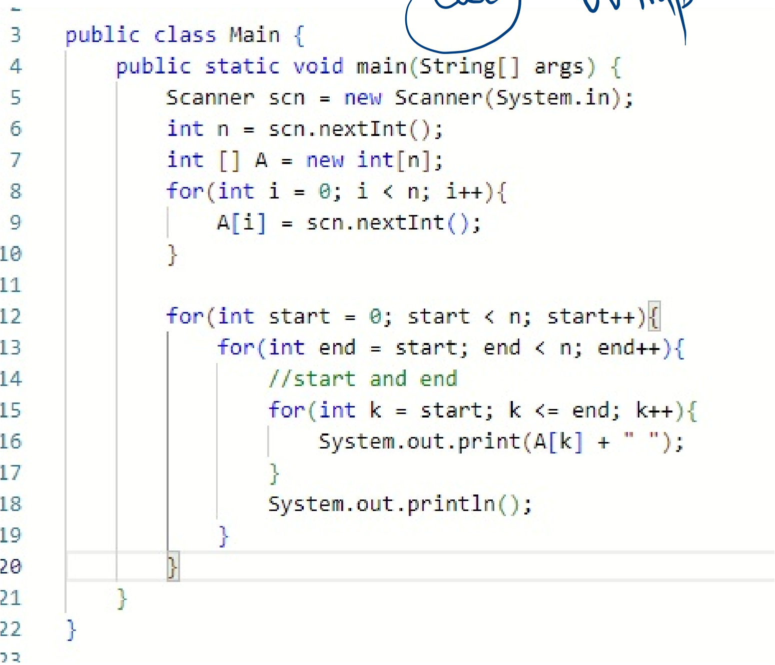
end = 1 2 3

start = 2

end. 2 3

Code

vv imp



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

        for(int start = 0; start < n; start++){
            for(int end = start; end < n; end++){
                //start and end
                for(int k = start; k <= end; k++){
                    System.out.print(A[k] + " ");
                }
                System.out.println();
            }
        }
    }
}
```

5	7	3	2
0	1	2	3

5

5 7

start = 0

end = ~~0~~ 1

0 < 4 ✓

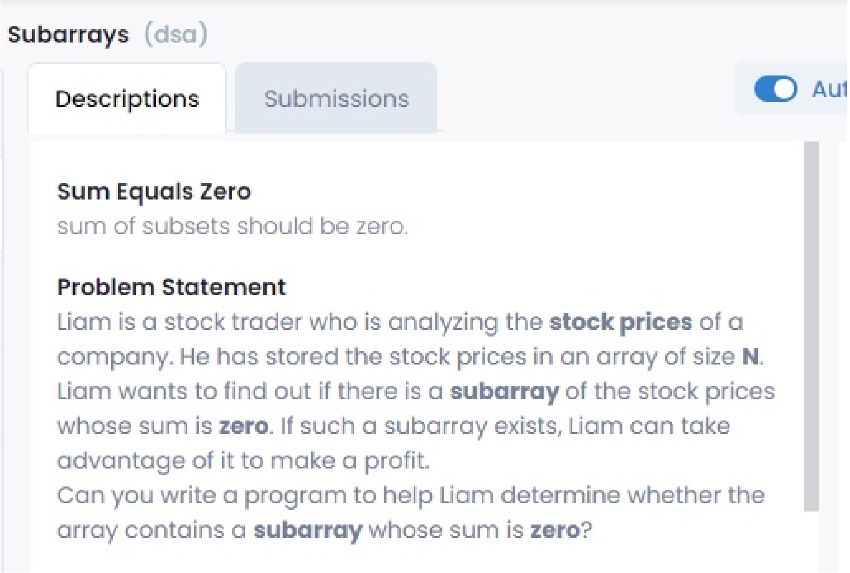
1 < 4 ✓

k = ~~0~~ ~~1~~ 2

k ≤ 1

1 ≤ 1 ✓

2 ≤ 1 ✗



Sum Equals Zero

sum of subsets should be zero.

Problem Statement

Liam is a stock trader who is analyzing the **stock prices** of a company. He has stored the stock prices in an array of size **N**. Liam wants to find out if there is a **subarray** of the stock prices whose sum is **zero**. If such a subarray exists, Liam can take advantage of it to make a profit.
Can you write a program to help Liam determine whether the array contains a **subarray** whose sum is **zero**?