Minimum difference 7

arr =

0	1	2	3
9	4	1	7

$K = 2$

pair	diff
9, 4	(5)
9, 1	(8)
9, 7	(2)
4, 1	(3)
4, 7	(3)
1, 7	(6)

mini of all diff

ans = 2

$K = 3$

	diff
9, 4, 1	(8)
9, 4, 7	(5)
4, 1, 7	(6)
9, 1, 7	(8)

ans = 5

Ex1

arr =

0	1	2	3
9	4	1	7

sort

arr =

0	1	2	3
1	4	7	9

K = 3

diff = ~~6~~ 5

Ex2

arr =

5	3	7	-2	-8	19	10	4	9	12	15	13

$K = 4$

size of window $= K = 4$

$n = 12$

arr =

0	1	2	3	4	5	6	7	8	9	10	11
-8	-2	3	4	5	7	9	10	12	13	15	19

(i) $(i+K-1)$

$$diff = arr[i+K-1] - arr[i]$$

$$diff = \cancel{12}\ \cancel{7}\ 4$$

first index $= 2 \longrightarrow (i)$

last index $= 5 \longrightarrow (i+K-1)$

Note:- Last index till we can travel for n as size of array & K as size of window is **n-k**.

psudo code

1) Sort the array

2) first index = i
   last index = i+k-1

   loop from 0 to n-k

   diff = arr[last index] − arr[first index]

   find min of diff

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 k = scn.nextInt();

    System.out.print(miniDiff(arr, n, k));
}
public static int miniDiff(int[] arr, int n, int k) {
    Arrays.sort(arr);
    int ans = Integer.MAX_VALUE;
    for (int i = 0; i <= n - k; i++) {
        int first = arr[i];
        int last = arr[i + k - 1];
        int diff = last - first;
        if ( diff < ans ) {
            ans = diff;
        }
    }
    return ans;
}
```

K = 3

i i+k-1

	0	1	2	3
arr =	1	4	7	9

ans = ~~+∞~~ ~~6~~ 5

diff = ~~6~~ 5

Form the largest number

$n = 4$

	0	1	2	3
arr =	4	46	8	9

46984 ≅ 46K

98446 ≅

✓ 98464 ≅

Logic

arr = (int)

0	1	2	3
4	46	8	9

Step I)

arr = (String)

0	1	2	3
"4"	"46"	"8"	"9"

// Convert int array into String array

Step II) (Imp)

a = "4"
b = "46"

"464" (b+a)

"446" (a+b)

// lambda function

arr = (String)

0	1	2	3
"9"	"8"	"46"	"4"

arr[i]

Step III)

String ans = "";

ans = "98464";

// convert String array into String

Note:-

Convert int to String

1) String str = String.valueOf(num);

2) String str = Integer.toString(num);

Note:-

$$\left.\begin{array}{l} a-b \\ b-a \end{array}\right\} \begin{array}{l} +ve \longrightarrow \downarrow ing \\ -ve \longrightarrow \uparrow ing \end{array}$$

Inbuilt fn

```
String str1 = ....
String str2 = ....

// return   str1.compareTo(str2);   ↑ing
return   str2.compareTo(str1);   ↓ing
```

arr = (String)

0	1	2	3
"9"	"8"	"46"	"4"

↑ arr[i]

String ans = "";

ans = "" + "9" = "9"

ans = "9" + "8" = "98"

ans = "98" + "46" = "9846"

ans = "9846" + "4" = "98464"

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(formLargestNumber(arr, n));
}
public static String formLargestNumber(int[] arr, int n) {
    // convert int array into string array
    String[] arr1 = new String[n];
    for (int i = 0; i < n; i++) {
        arr1[i] = Integer.toString( arr[i] );
    }

    // lambda function
    Arrays.sort(arr1, (a, b) -> {
        String str1 = a + b;
        String str2 = b + a;
        return str2.compareTo(str1); // str2 - str1 // decreadingg
        // return str1.compareTo(str2); // str1 - str2 // increadingf order
    });

    // convert string array into string
    String ans = "";
    for (int i = 0; i < n; i++) {
        ans = ans + arr1[i];
    }
    return ans;
}
```