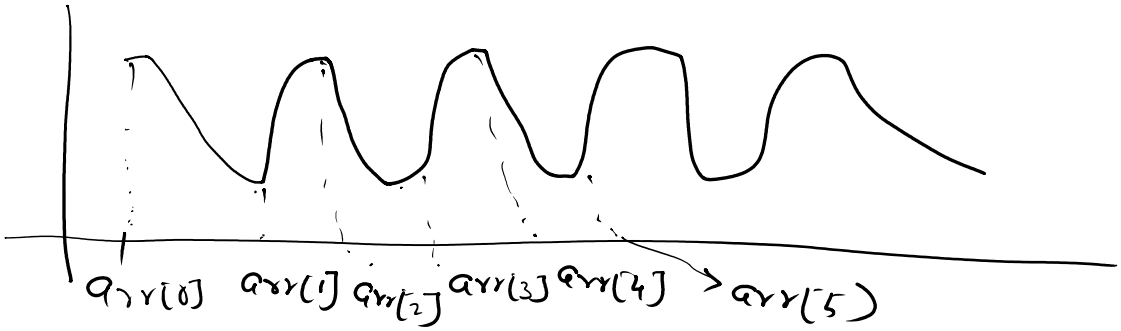
Sort an array in wave form 1



n = 7

arr = 10, 90, 49, 2, 1, 5, 23

Output - 2 1 10 5 49 23 90

Explanation of output.

After sorting the array

↳ 1, 2, 5, 10, 23, 49, 90

Swap the adjacent elements

↳ 2, 1, 10, 5, 49, 23, 90 → is it a wave?

↳ Yes

Step by step.

i = 0: 1, 2, 5, 10, 23, 49, 90 (swap i with i+1)

= 0 1 2 3 4 5 6

i = 2: 2, 1, 5, 10, 23, 49, 90 (swap i = 2 with i+1 = 3)

0 1 2 3 4 5 6

... 0 1 10 5 23 49 90 (swap i = 4 with i = 5)

i=4 2 1 10 5 23 49 90 (swap i=4 with i=5)

0 1 2 3 4 5 6

2 1 10 5 49 23 90 (wave form)

When no. of elements in array is even

i=0 1 2 5 10 23 49 90 95

i=2 2 1 5 10 23 49 90 95

i=4 2 1 10 5 23 49 90 95

i=6 2 1 10 5 49 23 90 95

2 1 10 5 49 23 95 90 → n=8

0 1 2 3 4 5 6 (7) ← n-1

↑ n-2

Code.

```
Arrays.sort(arr);
for(int i=0; i<n-1; i+=2) {
    int temp = arr[i+1];
    arr[i+1] = arr[i];
```

```
        arr[j] = temp;
    }
    for(int i=0; i<n; i++) {
        S.o.p(arr[j] + " ");
    }
```

Form the largest number

Input

4

4 46 8 9

Output

98464

Explanation:-

Example:- When we have only single digit element

4 6 8 1 9

0 1 2 3 4

1 4 + 6 + 8 + 9

$\times 10^0 + \times 10^1 + \times 10^2 + \times 10^3 + \times 10^4$

Problem arises when then it non-single digit element

4 46 8 9

4 8 9 46

↳ 46984 → not the largest

98464 → This is the largest

a="46", b="8".

a+b = 468 , (a+b) with (b+a)

✓ b+a = 846

4 46 8 9 → 46984

a=4

b=46

a+b = 446 } if (b+a > a+b){

b+a = 464 } swap(a,b);

}

46 4 8 9

a b → a+b=48
b+a=84

46 8 4 9

a b a+b=49
b+a=94

46 8 9 4

a b a+b=468
b+a=864

```
if(b+a>a+b){
swap(a,b);
```

8 46 9 4

a b a+b=469
b+a=946

b+a>a+b, swap(a,b)

8 9 46 4

a b a+b=469
b+a=9465

8 9 46 4

a b a+b=89
b+a=98

9 8 46 4

a b

Step 1.

↳ convert into String array

Step 2.

↳ Sort with customization

instead of comparing a,b
we will concantate a+b and b+a
then we will compare.

Code:-

```
String[] sarray = new String(n);
```

```
String[] sarray = new String[n];
for(int i=0; i<n; i++) {
  sarray[i] = String.valueOf(arr[i]);
  }
```

1st Way

```
Arrays.sort(arr, (a,b)->{ if(Integer.parseInt(b+a)>Integer.parseInt(a+b))
    return 1;
  else
    return -1;
});
```

2nd way

```
if ((b+a).compareTo(a+b)==1)
    return 1;
else
    return -1
```

3rd way

```
(b+a).compareTo(a+b),
```

```
Arrays.sort(arr, (b+a).compareTo(a+b));
```

1st method

```
//str[] = ["9", "8", "46", "4"]
for(int i=0; i<n; i++){
  S.o.P(str[i]); // 98464
  }
```

2nd method

```
String res="";
for(int i=0; i<n; i++) {
  rest = str[i];
  }
S.o.pln(res);
```

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be name
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        Integer arr[] = new Integer[n];
        for(int i=0;i<n;i++){
            arr[i] = sc.nextInt();
        }
        String sarr[] = new String[n];
        for(int i=0;i<n;i++){
            sarr[i] = String.valueOf(arr[i]);
        }
        Arrays.sort(sarr,(a,b)->(b+a).compareTo(a+b));
        for(int i=0;i<n;i++){
            System.out.print(sarr[i]);
        }
    }
}
```

Subarrays → it is set of consecutive elements in the array

i:0, j:0, k:i to j
j=1 k:i to j:0 to 1
j:2

Arr: [40, 60, 90, 5]

Subarrays →

40

40 60

40 60 90

40 60 90 5

60

60 90

60 90 5

90

90 5

5