swapMinMax1D

Write the C function `swapMinMax1D()` that takes in an array of integers *ar* and *size* (>1) as parameters, finds the index positions of the largest number and smallest number in the array, swaps the index positions of these two numbers, and passes the array to the calling function via call by reference. For example, if *ar* is `{1,2,3,4,5}`, then the resultant array *ar* will be `{5,2,3,4,1}` after executing the function. If there are more than one largest or smallest number in the array, we will swap the **last occurrence** of the largest and smallest numbers. For example, if *ar* is `{5,2,1,1,8,9,9}`, then the resultant array ar will be `{5,2,1,9,8,9,1}` after executing the function. The function prototype is:

```
void swapMinMax1D(int ar[], int size);
```

A sample program to test the function is given below:

```
#include <stdio.h>
void swapMinMax1D(int ar[], int size);
int main()
{
   int ar[50],i,size;

   printf("Enter array size: \n");
   scanf("%d", &size);
   printf("Enter %d data: \n", size);
   for (i=0; i<size; i++)
      scanf("%d",ar+i);
   swapMinMax1D(ar, size);
   printf("swapMinMax1D(): ");
   for (i=0; i<size; i++)
      printf("%d ",*(ar+i));
   return 0;
}
void swapMinMax1D(int ar[], int size)
{
  /* Write your code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter array size:
5
Enter 5 data:
1 2 3 4 5
swapMinMax1D(): 5 2 3 4 1
```

(2) Test Case 2:

```
Enter array size:
2
Enter 2 data:
5 5
swapMinMax1D(): 5 5
```

(3) Test Case 3:

```
Enter array size:
7
Enter 7 data:
1 1 1 5 5 5 5
swapMinMax1D(): 1 1 5 5 5 5 1
```

(4) Test Case 4:

```
Enter array size:
```

```
9
Enter 9 data:
9 1 1 9 9 5 5 5 5
swapMinMax1D(): 9 1 9 9 1 5 5 5 5
```