Lab: Dynamic Array

Write a C++ program that does the following:

1. Prompt the user for the number of values to be used (say **number_of_values** = *50*], the maximum (integer) limit, and the minimum (integer) limit.
2. Dynamically allocate an array exactly large enough to hold that many integer values. Use the following syntax to create a dynamic array:
   - `int * array_values = new int[number_of_values]; /* now we can use array_values as if it is an integer array. The size of the array is: number_of_values. */`

3. Fill the array with randomly generated values between the user-specified minimum and maximum limits, inclusive. Use the provided function in the template file.
4. Sort the array in ascending order. You may use the supplied selection sort function. However, modify it so that you can use the function to sort array variables.
5. Print to the screen:
   A. all the values in ascending order and in descending order
   B. the smallest value in the array
   C. the largest value in the array

B. Before the program ends (before return 0 in the main function), delete the array_values from memory. You can use the following syntax to delete an array:
   - `delete [] array_values;`

Sample run #1:

```
Enter the number of values: 10
Minimum limit: 1
Maximum limit: 20

The values in order: [1, 1, 4, 4, 7, 7, 8, 8, 18, 18]
The values in reverse order: [18, 18, 8, 8, 7, 7, 4, 4, 1, 1]
The smallest value in the array is 1
The largest value in the array is 18
```

Sample run #2:

```
Enter the number of values: 10
Minimum limit: 1
Maximum limit: 15

The values in order: [1, 1, 1, 2, 3, 5, 12, 13, 13, 15]
The values in reverse order: [15, 13, 13, 12, 5, 3, 2, 1, 1, 1]
The smallest value in the array is 1
The largest value in the array is 15
```

At a minimum you must implement functions to

1. fill the array with random values,
2. sort the array, and to
3. compute the maximum/minimum of the values in the array. You may define other functions if it is required in the program.

Please study how we are dynamically allocating and de-allocating memory for the array in this program to gain more insight into pointers in C++.

When you are satisfied with your program, submit it by the due date of 5pm Friday, 1 May.