B) Correctness of Algorithm

- At the moment that we start to run the program by calling any of the functions (worst, average or best), it is 100% sure that it will work as we have the *array[size]* and *int i = 0* and the function is also correct (for sorting).
- The first *for* loop is to visit all the elements of the array and the second *for* loop is to visit all the elements expect the one visited before. The *if* is to compare the element of the first *for* loop with the element of the second *for* loop. In the end we just swap the positions (smaller-larger) and print the sorted elements.
- The loop will always terminate when the *i* gets bigger then *size* (which will not happened).

C) Worst, Average and Best case of the Algorithm

- Worst case: The elements are in reverse order so each one them has to move/change position in the array.
  i.e.: [9, 8, 7, 6, 5, 4, 3, 2, 1] -> [1, 2, 3, 4, 5, 6, 7, 8, 9]
- Average case: The elements are random. I use a C++ function to generate random inputs *rand()* for each time of the execution of the program *srand(time(0)).*
  i.e.: [4, 7, 1, 2, 8, 9, 3, 5, 6] -> [1, 2, 3, 4, 5, 6, 7, 8, 9]
- Best case: The elements are already sorted. In this case I decided to just print them from 0 to the nr. of *size-1*.
  i.e.: [1, 2, 3, 4, 5, 6, 7, 8, 9] -> [1, 2, 3, 4, 5, 6, 7, 8, 9]