1 Bubble Sort in C

Bubble Sort is the probably the first sorting algorithm everyone is introduced to. Let's first include the necessary header files:

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
```

The main operation of Bubble Sort is swapping two values:

```
void swap(int *x, int *y)
{
    int t = *x;
    *x = *y;
    *y = t;
}
```

The easiest way to remember how to implement it is when you assign something to the variable `t` like `int t = *x;` you are saving that thing so on the next line you can override it `*x = *y;`. And after that finish the process by assigning the second variable `*y = t;`.

Alright, now let's implement the bubble sort itself:

```
void bubble_sort(int *xs, int xs_size)
{
   for (int i = xs_size - 1; i > 1; --i) {
       for (int j = 0; j < i; ++j) {
           if (xs[j] > xs[j + 1]) {
               swap(&xs[j], &xs[j + 1]);
           }
       }
   }
}
```

Let's check if this bubble sort algorithm works correctly. Let's implement a function that can generate `n` random numbers for testing.

```
#define MAX_X_SIZE 100

void generate_n_numbers(int *xs, int n)
{
    for (int i = 0; i < n; ++i) {
        xs[i] = rand() % MAX_X_SIZE;
    }
}
```

We also need to be able to check that the array is sorted. We can do that by iterating the array with a "window" of size 2 and checking if the pairs are ascending

```
// 1 2 3 5 4 6
// ^ ^ ascending
// 1 2 3 5 4 6
//   ^ ^ ascending
// 1 2 3 5 4 6
//     ^ ^ ascending
// 1 2 3 5 4 6
//       ^ ^ DESCENDNIG!!!
bool is_sorted(int *xs, int n)
{
    for (int i = 0; i < n - 1; ++i) {
        if (xs[i] > xs[i + 1]) {
            return false;
        }
    }
    return true;
}
int main(){}
```