Counting Sort

- makes assumptions about the data
- doesn't use comparison
- counts the number of occurrences of each value
- only works with non-negative discrete values (can't work with floats, strings)
- values must be within a specific range
- we won't use it to sort elements within a range of 1 to 1 000 000
- NOT an in-place algorithm
- O(n) – can achieve this because we're making assumptions about the data we're sorting
- unstable
- if we want the sort to be stable we have to do some extra steps

Example:

1. Assume we have values between 1 and 10 inclusive
2. We have 10 possible values, so we create a counting array of length 10
3. Traverse the input array from left to right
4. Use the counting array to track how many of each value are in the input array
5. Using counts in the counting array, write the values in sorted order to the input array

STEP	2	5	9	8	2	8	7	10	4	3	DESC
0	++										
1		++									
2			++								
3				++							
4					++						
5						++					
6							++				
7								++			
8									++		
9										++	
10	2	2	3	4	5	7	8	8	9	10	

Writing value back to original input array checking one by one how many time each value we have

Counting array

STEP	1	2	3	4	5	6	7	8	9	10
0	0	+1	0	0	0	0	0	0	0	0
1	0	0	0	0	+1	0	0	0	0	0

2	0	0	0	0	0	0	0	0	+1	0
3	0	0	0	0	0	0	0	+1	0	0
4	0	+1	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	+1	0	0
6	0	0	0	0	0	0	+1	0	0	0
7	0	0	0	0	0	0	0	0	0	+1
8	0	0	0	+1	0	0	0	0	0	0
9	0	0	+1	0	0	0	0	0	0	0
	0	2	1	1	1	0	1	2	1	1