Exercise: Functions for sorting in `numpy`

For sizes $N = 2^0, 2^1, \ldots, 2^12$ and plot the CPU times of `bubblesort` and various sorting algorithms `quicksort`, `mergesort` or (which is the same as `stable` or `timsort`), `heapsort` using sorting the functions available in `numpy`. According to `numpy`'s manual https://numpy.org/doc/stable/reference/generated/numpy.sort.html, the fastest algorithm is `heapsort`.

```
import numpy as np
import time

start_time = time.time()
sorted_array = bubble_sort(array)
end_time = time.time()
time_bubble = end_time - start_time

start_time = time.time()
sorted_array = np.sort(array, kind='quicksort')
end_time = time.time()
time_quicksort = end_time - start_time

start_time = time.time()
sorted_array = np.sort(array, kind='mergesort')
end_time = time.time()
time_mergesort = end_time - start_time

start_time = time.time()
sorted_array = np.sort(array, kind='heapsort')
end_time = time.time()
time_heapsort = end_time - start_time
```

Use the code above and the code given in the classroom to plot the time taken from all four algorithms. In the same figure, plot aN^2 and $bN\log(N)$, where a and b are adjustable constants to make the curves fit in the graph.