Implementation of Josephus using list

So firstly, we initialize a Q by using an empty list Q, and a for loop to place all the element in correct order. We also have another empty list to store our results. The for loop is to pop out the elements of the Q from the first element to the (M-1)th element and respectively put them back to the end of the Q. SO after that, the Mth element we need to find will be place at the first position of the Q. So we pop that out and append it to the results list. The operations carries on until all the element in the Q has been popped out or the Q is empty.

Implementation of Josephus using deque: 20 points

Same operations, but using a list for a Q, we use a deque imported from collections. And to pop out the first element, we use popleft function.

Analysis of timings: 10 points

As you can see from the table, as the size increase, the time consumed of the List version got larger and larger, so Deque version is faster. This is because Deque is actually a doubly linked-list, so it is faster to do operations like append or pop the head or tail of the Q because it uses pointers.