Q1. Determining the end of an array.

A 🡪

Algorithm:

|  |
| --- |
| 1. Start checking the elements at index two to the power ‘i’, that is 2, 4, 8, 16, 32, 64, …, 2i 2. Stop once you have found the first occurrence of ‘Infinity’. Suppose you stop at an index ‘k’ 3. Now binary search a number that is smaller than ‘Infinity’, between values 2k and 2k-1 4. Once you found that number start moving one step at a time and find first occurrence of ‘Infinity’ |

Complexity:

1. As we are moving in the range of power of 2, and there are total ‘n’ non-infinity elements, first step of algorithm would take log(n) elements.
2. Second step which requires comparison can be done in constant time
3. In third step as we do a binary search between n to n/2 elements, the running time would be log(n/2)
4. In last step algorithm checks element one by one thus, at max will have to check log(n/2) elements.
5. Thus, total complexity would be

f(n) = log(n) + log(n/2) + log(n/2)

f(n) = O(log(n)) … (as log(n/2) < log(n))