```
/**
sequences are a readonly arrays in dafny, not efficient but proving w/ sequences is easy
 */
method {:verify false} Search(q: seq<int>, key: int) returns (i: nat)
    requires key in q //syntactic suger for (exists i :: 0 ≤ i < |q| && q[i] == key)
    ensures i < |q| && q[i] == key
    {
        i := 0;
        while q[i] ≠ key
            invariant i < |q| && key in q[i..] //q[i...] == the suffix of the array from i
            decreases |q| - i
        {
            assert i < |q| && key in q[i..];
            assert q[i] ≠ key;
            // ⟹ ? NOT before adding the "key in q[i.." to the invariant: (counter example : q=[1],i=0,key = 2, q = [2,3], i = 1, key = 2).
            // Although it can't happen, the invariant doesn't know the logic before. So we need to amplify the invriant.
            assert i+1 < |q| && key in q[i+1..];
             i:= i+1;
             assert i < |q| && key in q[i..];
        }
    }
```