Algorithm 1 Insertion Sort(Non-increasing order)

```
1  for j = 2 to A.length
2      key = A[j];
3      i = j − 1;
4      while (j > 0 and A[i] < key)
5          A[i + 1] = A[i];
6          i − −;
7      A[i + 1] = key;
```

Algorithm 2 Linear search

Input: A sequence of n numbers A and a constant v

Output: Index i;

```
1  for i = 1 to A.length
2      if A[i] == v
3          return i;
4  retunr NIL;
```

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ T(n-1) + n - 1 & \text{if } n > 1. \end{cases}$$

Algorithm 3 Binary adding algorithm

Input: A sequence of n numbers A and a sequence of n numbers B

Output: A sequence of $n+1$ numbers $C = [c_1, c_2, ..., c_{n+1}]$;

```
1  for i = 1 to n
2        C[i] = (A[i] + B[i] + carry)%2;
3        carry = (A[i] + B[i] + carry)/2
4  C[i + 1] = carry;
```

Algorithm 4 Selection Sort

Input: A sequence of n numbers unsorted array A;

Output: Sorted array A;

```
1  for i = 1 to n − 1
2        min = i;
3        for j = i + 1 to n
4              if A[j] < A[min]
5                    min = j;
6        A[i] = temp;
7        A[i] = A[min];
8        A[min] = A[i];
```

Algorithm 5 Merge(A,p,q,r)

```
1  n_1 = q − p + 1;
2  n_2 = r − q;
3  Let L[1..n_1] and R[1..n_2] be new arrays
4  for i = 1 to n_1
5        L[i] = A[p + i − 1];
6  for j = 1 to n_2
7        R[j] = A[q + i];
8  i = 1;
9  j = 1;
10 k = p;
11 while i < n_1 and j < n_2
12       if L[i] ≤ R[j]
13             A[k] = L[i];
14             i + +;
15       else
16             A[k] = R[j];
17             j + +;
18       k + +;
19 if j == n_2
20       for m = k to r
21             A[m] = L[i];
22             i + +;
```

Algorithm 6 BinarySearch(A,target)

```
1  low = 1;
2  high = A.length;
3  while low <= high
4        mid = (low + high)/2;
5        if A[mid] == target
6              return mid;
7        if A[mid] > target
8              high = mid − 1;
9        else
10             low = mid + 1;
11 return −1;
```