b. selection sort

```
1,      for i =0 to n-1 do :
2,      minIndex = i
3,          for j =i+1 to n do:
4,              if A[j] < A[minIndex]:
5,                  minIndex = j
6,              end if
7,          end for
8,          if list[minIndex] < list[i]:
9,              exchange A[minIndex] with A[i]
10,     end for
```

Loop invariants:

Initialization:

We start by showing that the loop invariant holds before the first loop iteration, when i=0, we assume the index of minimum element is 0, then we go into the second for loop starts at line 4, in this for loop, we iterate all the element from index 1(i+1) to the last one to check if the smallest element in the subarray A[1….last one], if the smallest element of subarray smaller than list [i], then exchange the value of these two, that move the smallest element of the subarray to the current index.

Maintain:

After finding the smallest element of the subarray and exchanging the smallest element and current element, the algorithm increase i which makes i= i+1, and then find the smallest element of the subarray from list[i+2] to list[list.length-1], then it compares the smallest element of the subarray with the current element which is A[i+1] to make sure it put the smallest element from subarray into A[i+1], so the algorithm remains true for the next iterations.

Termination:

When the loop terminates, the element in the last position (at the end of the array) is the biggest element of the whole array- list. Which i = list.length-1. Now we have the subarray list[0..list.length-2] sorted and list[list.length-1] which is the biggest element of the array list. Since it doesn't been selected from the previous iterations. Therefore, we conclude that the entire array is sorted. Hence the algorithm is correct