Problem 9.2 (a)

i , i	1	2	3
S_i	0	2	3
f_i	3	4	6
duration	3	2	3

This approach selects just *i2*, but the optimal solution selects *i1, i3*

Also;

Say (*n-1*) disjoint low-duration intervals that overlap throughout (*n*) will result to higher duration intervals.

```
        |===|                   1
             |===|              2
|=====|                          *
        |====|                  *
               |======|          *
```

- In those 5 tasks listed above, the optimal set of activities are those labeled (*).
- The ones with the least duration are obviously the (1) & (2), but this technique selects only two in this example instead of the optimal three.

With that said, a greedy algorithm for the activity-selection that makes greedy choice of selecting the activity with the shortest duration may fail.

Reference: http://mypathtothe4.blogspot.com/2013/03/greedy-algorithms-activity-selection.html

Problem 9.2 (b)

Another solution except the one that we have to sort and then select the activity would be an implementation of brute-force; by which I mean first select and then check if it fits with the requirements, if not try with next element and so on.

Pseudocode:

```
i = 0
for j in range size // where size is the number of activities
        if finish[j] >= start[]
// If finish time is greater then the starting time of previous activity
                i = j
```