1. (Reading). Done

2. (Finding the max, 10pt)

a) [5pt] Write a program which finds the maximum element in a list.

My code is below:

```
def find_max(lst):
    i = 0
    mx = float('inf') #negative infinity
    #max == max(lst[0:i])
    while (i < len(lst)):
        mx == max(lst[0:i])
        if lst[i] > mx:
            mx = lst[i]
            i += 1
    #mx == max(lst[0:i]) and i == len(lst), so
    #max == max(lst[0:i]) == max(lst[0:len(lst)]) = max(lst)
    return mx
```

And here is a testrun; output looks correct based on testrun.

```
>>> from random import random
>>> lst = [random() for i in range(0,10)]
>>> find_max(lst)
0.9815636136897965
```

b) [5pt] What is the loop invariant for your code, and show that it is correct.

I included the loop invariant in my screenshot above. It is:

mx = max(lst[0:i])
so anytime the loop starts, or restarts, the variable mx contains the maximum of the list lst from positions 0 through i-1. Initially, this is true, since i = 0, and the list is empty, so both values are negative infinity. After each run through the body of the loop, the invariant still holds, since i is increased by one, and the element at position i in lst is compared to mx, and the larger one is retained. Eventually, when the loop terminates, i = lst[i], so mx = max(lst).

c) [5pt] Does the loop terminate?

Yes, it does. In each step i is increased until it has reached len(lst). This will happen after at most len(lst) steps. So the algorithm terminates and is correct.

3. [...] ....