CSCI 3104 Assignment 2

10:00 - 10:50 Wanshan

Jackson Chen

9 February 2016

1. (a)

```
max = a[0]
min = a[0]
for (i in a) {
  if (a[i] > max) {
    max = a[i]
  }
  if (a[i] < min) {
    min = a[i]
  }
}
return max, min
```

   (b)

```
function findExtremas(array a, int start, int end) {
  // Base case
  if (start - end < 2) {
    return a[start], a[start] // max, min
  }
  int mid = (start+end)/2
  int u = findExtremas(a, start, m)
  int v = findExtremas(a, m+1, end)

  if (u > v) {
    return u, v //max, min
  }
  else {
    return v, u //max, min
  }
}
```

   (c)

$$T(n) = 2T(\frac{n}{2}) + O(1)$$

(d)

$$\text{Since } f(n) = O(n^{\log_2 2-1})$$

$$T(n) = \Theta(n^1) = \boxed{\Theta(n)} \text{ by the Master method}$$

2. (a) There is no majority element since the majority elements in the two subarrays are different. Therefore it is impossible for there to be another element with $\frac{n}{2}+1$ frequency.

(b)

```
function majorEl(array a) {
  // Base case(s)
  if (a.length == 1) {
    return a[0]
  }

  mid = a.length/2
  l1 as array = a[0] ... a[mid]
  l2 as array = a[mid+1] ... a[n-1]

  e1 = majorEl(l1)
  e2 = majorEl(l2)

  if (e1 == e2) {
    return e1
  }
  else {
    if (getFrequency(a, e1) > mid + 1)
      return e1
    else if (getFrequency(a, e2) > mid + 1)
      return e2
    else
      return NULL
  }
}
```

(c)

$$T(n) = 2T(\frac{n}{2}) + O(n)$$

$$\text{Since } O(n) = O(n^{\log_2 2})$$

$$T(n) = \boxed{\Theta(n \log n)} \text{ by the Master method}$$