UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

COS 212 Tutorial 9: Version B

- 03/05/2012
- 50minutes
- 3 questions for a total of 21 marks.

Name:_____

Student/staff Nr:_____

Marker (office use):_____

Question 1 Insertion Sort: ..... (7 marks)

1.1 Suppose an array containing n elements initially. Comment on the following statement: (2)

`If the number of elements is tripled, then the effort of the algorithm will increase by a factor of 9.`

> **Solution:** The agorithms' complexity is calculated in terms of the number of comparisons which is in turn determined by the number of elements. Insertion sort is an $O(n^2)$ algorithm so the effort doubles for each doubling of the number of elements, so if $3n$ then $\frac{((3n)^2)}{9} = n^2$

Answer:

1.2 Consider the insertion sort algorithm:

```
insertion(arr[])
 for(i = 1; i < arr.length; ++i)
   el = arr[i]
   for(j = i; j > 0 && el < arr[j-1]; --j)
     arr[j] = arr[j-1]
   arr[j] = el
```

a) In what regard does this version differ from the algorithm in the textbook and what is the implication of this? Discuss your answer thouroughly. (3)

> **Solution:** I made a mistake in this question, give full marks.

Answer:

b) Why would shifting of the elements in the array cease when no smaller element than `el` is found. Refer to why the condition `el < arr[j-1]` forms part of the stopping condition of the inner `for`-loop. (2)

> **Solution:** because from that point forward everything will be smaller than el. The mechanics of the algorithm finds the correct place to insert an element starting from 1 side of the array. Therefore, the first element will be placed in its correct position, the next will be placed correctly relative to the first etc.

Answer:

Question 2 Selection Sort: ..... (11 marks)

2.1 Compare the insertion sort and selection sort algorithms. (3)

> **Solution:** Insertion sort moves/shifts multiple elements in each iteration of the algorithm while selection swaps only two in each iteration. Both find the final placements of some current element in each iteration.

Answer:

2.2 Consider the array `[50,1,10,9,8,7,6,100,30,20]`. Apply selection sort to this array and redraw the array after `every` swap operation. (8)

Answer:

Solution: 1 mark for each correct swap and 1 additional mark if the final array is sorted.

[1,50,10,9,8,7,6,100,30,20] // 50 and 1 swapped

[1,6,10,9,8,7,50,100,30,20] // 50 and 6 swapped

[1,6,7,9,8,10,50,100,30,20] // 10 and 7 swapped

[1,6,7,8,9,10,50,100,30,20] // 8 and 9 swapped

[1,6,7,8,9,10,20,100,30,50] // 50 and 20 swapped

[1,6,7,8,9,10,20,30,100,50] // 30 and 100 swapped

[1,6,7,8,9,10,20,30,50,100] // 100 and 50 swapped

Question 3 Bubble Sort: ..... (3 marks)

3.1 Consider the following suggested alternative for the bubble sort algorithm: (3)

```
bubble(arr[])
 endIndex = arr.length
 repeat
   exchanged = false
   for(i = 1; i < endIndex; ++i)
   {
     if(arr[i] < arr[i-1])
       swap(arr[i],arr[i-1])
       exchanged = true
   }
   endIndex--
 until !exchanged
```

You may assume that swap correctly exchanges `arr[i]` and `arr[i-1]`.

Discuss how the flag `exchanged` is used to terminate the algorithm when no further execution is required. Discuss your answer in terms of how `endIndex` and `exchanged` relate.to one another.

Solution: Exchanged indicates that there was at least one swap during the iteration of the for. Once a swaps have occurred one element has found its final place in the array by "bubbling" left or right (either largest or smallest). No assumption can be made about the other elements' placements though. As soon as no swap occurs it implies that all elements were found to be in their correct places.

EndIndex is used to discard an already sorted portion of the array, everything after endIndex is sorted and thus no exchanges will be made from this point onwards.

Answer: