

2018 SS

Question 37

[1 mark] What is the output of the following code?

```
x = [6, 3, 0, 9, 1, 6, 5, 4]
y = sorted(x)
print('x =', x, 'y =', y)
```

- (a) x = [6, 3, 0, 9, 1, 6, 5, 4] y = [9, 6, 6, 5, 4, 3, 1, 0]
- (b) x = [6, 3, 0, 9, 1, 6, 5, 4] y = [0, 1, 3, 4, 5, 6, 6, 9]
- (c) x = [0, 1, 3, 4, 5, 6, 6, 9] y = [0, 1, 3, 4, 5, 6, 6, 9]
- (d) x = [9, 6, 6, 5, 4, 3, 1, 0] y = [9, 6, 6, 5, 4, 3, 1, 0]
- (e) None of the above

Question 38

[1 mark] What is the Big O time complexity of the python sorted() function?

- (a) $O(\log n)$
- (b) $O(n^2)$
- (c) $O(n \log n)$
- (d) $O(n)$
- (e) None of the above

Given the following function definitions used in a bubble sort algorithm:

```
def swap(values, i, j):
    values[i], values[j] = values[j], values[i]
def bubble(values):
    for i in range(len(values) - 1):
        if values[i] > values[i + 1]:
            swap(values, i, i+1)
def bubble_sort(values):
    for i in range(len(values)):
        bubble(values)
```

Question 39

[1.5 marks] What will the following list look like after **three** passes of the bubble operation?

[54, 26, 93, 17, 77, 31, 44, 55, 20]

- (a) [17, 26, 31, 44, 20, 54, 55, 77, 93]
- (b) [17, 26, 31, 44, 54, 20, 55, 77, 93]
- (c) [54, 26, 17, 31, 44, 20, 55, 77, 93]
- (d) [26, 17, 31, 44, 20, 54, 55, 77, 93]
- (e) None of the above

Question 40

[1 mark] What is the Big-O time complexity for sorting a list using the bubble sort algorithm?

- (a) $O(\log n)$
- (b) $O(n \log n)$
- (c) $O(n^2)$
- (d) $O(n)$
- (e) None of the above

Given the following function definitions used in a selection sort algorithm:

```
def swap(values, i, j):
    values[i], values[j] = values[j], values[i]
def selection_sort(values):
    for fill_slot in range(len(values)-1, 0, -1):
        pos_max = 0
        for i in range(1, fill_slot+1):
            if values[i] > values[pos_max]:
                pos_max = i
        swap(values, pos_max, fill_slot)
```

Question 41

[1.5 marks] What will the following list look like after **three** passes of the selection sort algorithm?

[29, 10, 14, 37, 13]

- (a) [13, 14, 10, 29, 37]
- (b) [13, 10, 14, 29, 37]
- (c) [37, 29, 14, 10, 13]
- (d) [10, 14, 13, 29, 37]
- (e) None of the above

Question 42

[1 mark] What is the Big O time complexity for sorting a list using the selection sort algorithm?

- (a) $O(n \log n)$
- (b) $O(n)$
- (c) $O(\log n)$
- (d) $O(n^2)$
- (e) None of the above

2017 S2

Question 27

[2 marks] Given the following list: [93, 17, 26, 31, 77], what does the list look like after the **THIRD** pass when sorting it in ascending (increasing) order using the **Insertion Sort** algorithm as discussed in the lectures?

- (a) [17, 26, 93, 31, 77]
- (b) [17, 26, 31, 93, 77]
- (c) [93, 17, 26, 31, 77]
- (d) [17, 26, 31, 77, 93]
- (e) None of the above.

Question 29

[2 marks] Given is a list with the elements $[2k, 2k-1]$ for $k=1, \dots, n/2$ (e.g. for $n=10$ the list is [2,1,4,3,6,5,8,7,10,9]). Which of the following statements is true?

- (a) Sorting the above list with Bubble Sort has a time complexity (running time) of $O(n \log n)$
- (b) Sorting the above list with Insertion Sort has a time complexity (running time) of $O(n)$
- (c) Sorting the above list with Selection Sort has a time complexity (running time) of $O(n)$
- (d) Sorting the above list with Selection Sort has a time complexity (running time) of $O(n \log n)$
- (e) Sorting the above list with Bubble Sort has a time complexity (running time) of $O(n)$

Question 38

Consider the list of numbers below:

[50, 8, 57, 6, 90, 17, 89]

Using the techniques demonstrated in lectures, show the list for each pass when using:

a) Selection sort

b) Bubble sort

(5 marks)

2017 S1

2 marks] Given a list with the values 1 to n in reverse order, i.e. $[n, n-1, \dots, 3, 2, 1]$, which of the following five statements are true?

- I Sorting the above list with Insertion Sort has a time complexity (running time) of $O(n^2)$
- II Sorting the above list with Bubble Sort has a time complexity (running time) of $O(n^2)$
- III Sorting the above list with Selection Sort has a time complexity (running time) of $O(n^2)$
- IV Sorting the above list with Shell Sort has a time complexity (running time) of $O(n)$
- V Sorting the above list with Merge Sort has a time complexity (running time) of $O(n \log n)$

- (a) II, III and V
- (b) I, II, III, and V
- (c) I, II and III
- (d) II and V
- (e) All statements I-V are true

2017 S1

Question 40

[2 marks] Given the list [94, 22, 52, 41, 7, 76, 18, 39], what does the list look like after the first pass when sorting it in ascending (increasing) order using the Bubble Sort algorithm discussed in the lectures?

- (a) [22, 52, 41, 7, 76, 18, 39, 94]
- (b) [7, 22, 18, 39, 94, 76, 52, 41]
- (c) [22, 94, 52, 41, 7, 76, 18, 39]
- (d) [39, 22, 52, 41, 7, 76, 18, 94]
- (e) None of the above.

Question 41

[2 marks] Given the list [94, 22, 52, 41, 7, 76, 18, 39], what does the list look like after the first pass when sorting it in ascending (increasing) order using the Selection Sort algorithm discussed in the lectures?

- (a) [22, 52, 41, 7, 76, 18, 39, 94]
- (b) [7, 22, 18, 39, 94, 76, 52, 41]
- (c) [39, 22, 52, 41, 7, 76, 18, 94]
- (d) [22, 94, 52, 41, 7, 76, 18, 39]
- (e) None of the above.

Question 42

[2 marks] Given the list [94, 22, 52, 41, 7, 76, 18, 39], what does the list look like after the first pass when sorting it in ascending (increasing) order using the Insertion Sort algorithm discussed in the lectures?

- (a) [7, 22, 18, 39, 94, 76, 52, 41]
- (b) [39, 22, 52, 41, 7, 76, 18, 94]
- (c) [22, 52, 41, 7, 76, 18, 39, 94]
- (d) [22, 94, 52, 41, 7, 76, 18, 39]
- (e) None of the above.

Question 44

[2 marks] Below is an implementation of the Insertion Sort algorithm (for sorting a list in **descending** (decreasing) order). Three lines of the code have been omitted.

```
def my_insertion_sort(a_list):
    for index_number in range(1, len(a_list)):
        item_to_insert = a_list[index_number]
        index = index_number - 1
        #DELETED_LINE_1
        #DELETED_LINE_2
        #DELETED_LINE_3
        a_list[index + 1] = item_to_insert
```

What code do we need to insert in the three deleted lines in order to get Insertion Sort algorithm correctly sorting a list in **descending** order?

- (a) while index >= 0 and a_list[index] < item_to_insert:
 a_list[index + 1] = a_list[index]
 index -= 1
- (b) while index >= 0 and a_list[index] > item_to_insert:
 a_list[index + 1] = a_list[index]
 index -= 1
- (c) while index >= 0 and a_list[index] < item_to_insert:
 a_list[index] = a_list[index + 1]
 index -= 1
- (d) while index >= 0 and a_list[index] > item_to_insert:
 a_list[index] = a_list[index + 1]
 index -= 1

(e) None of the above.

Question 54

[2 marks] Given a list with the values 1 to n in reverse order, i.e. [n, n-1, ..., 3, 2, 1], which of the following five statements are true?

- I Sorting the above list with Insertion Sort has a time complexity (running time) of $O(n^2)$
- II Sorting the above list with Bubble Sort has a time complexity (running time) of $O(n^2)$
- III Sorting the above list with Selection Sort has a time complexity (running time) of $O(n^2)$
- IV Sorting the above list with Shell Sort has a time complexity (running time) of $O(n)$
- V Sorting the above list with Merge Sort has a time complexity (running time) of $O(n \log n)$

- (a) II, III and V
- (b) I, II, III, and V
- (c) I, II and III
- (d) II and V
- (e) All statements I-V are true

2017 SS

Question 36

[8 marks]

Consider the `insertion_sort()` function as defined below:

```
def insertion_sort(values):  
    for index in range(1, len(values)):  
        current_value = values[index]  
        position = index  
        while position > 0 and values[position - 1] > current_value:  
            values[position] = values[position - 1]  
            position = position - 1  
        values[position] = current_value  
        print(index, ': ', values)
```

Notice that a `print()` statement has been added at the very end of the `insertion_sort()` function. This is so that we can visualise the order of the elements in the list at the *end* of each iteration of the insertion sort algorithm. The `print()` statement displays the value of the variable `index`, as well as the list items.

If the `insertion_sort()` function is called with the following list:

```
my_list = [4, 3, 2, 1]  
insertion_sort(my_list)
```

then the `for` loop in the `insertion_sort()` function will be executed exactly 3 times (the value of the variable `index` will change from 1 to 2 to 3 as this loop executes). Exactly what would be printed by the `print()` statement (at the end of each iteration of the `for` loop) as the `insertion_sort()` function executes?

The last line of output (when the array is sorted) is provided, as are the values of `index`. Complete the missing two lines of output below:

```
1 : [          ]  
2 : [          ]  
3 : [ 1 , 2 , 3 , 4 ]
```

2016S2

Question 36

[2 marks] Given is the list [48, 64, 11, 66, 52, 31, 40, 41]. What does the list look like after the first pass when sorting it in ascending (increasing) order with the Bubble Sort algorithm discussed in lectures?

- (a) [48, 11, 64, 52, 31, 40, 41, 66]
- (b) [48, 31, 11, 41, 52, 64, 40, 66]
- (c) [48, 64, 11, 66, 52, 31, 40, 41]
- (d) [48, 64, 11, 41, 52, 31, 40, 66]
- (e) None of the other answers.

Question 37

[2 marks] Given is the list [48, 64, 11, 66, 52, 31, 40, 41]. What does the list look like after the first pass when sorting it in ascending (increasing) order with the Selection Sort algorithm discussed in lectures?

- (a) [48, 64, 11, 66, 52, 31, 40, 41]
- (b) [48, 64, 11, 41, 52, 31, 40, 66]
- (c) [48, 11, 64, 52, 31, 40, 41, 66]
- (d) [48, 31, 11, 41, 52, 64, 40, 66]
- (e) None of the other answers.

Question 38

[2 marks] Given is the list [48, 64, 11, 66, 52, 31, 40, 41]. What does the list look like after the first iteration (gap size 4) when sorting it in ascending (increasing) order with the Shell Sort algorithm discussed in lectures?

- (a) [48, 31, 11, 41, 52, 64, 40, 66]
- (b) [48, 64, 11, 66, 52, 31, 40, 41]
- (c) [48, 64, 11, 41, 52, 31, 40, 66]
- (d) [48, 11, 64, 52, 31, 40, 41, 66]
- (e) None of the other answers.

Question 39

[2 marks] Given is a list with the elements $[2k, 2k-1]$ for $k=1, \dots, n/2$ (e.g. for $n=10$ the list is $[2,1,4,3,6,5,8,7,10,9]$). Which of the following statements is true?

- (a) Sorting the above list with Selection Sort has a time complexity (running time) of $O(n \log n)$
- (b) Sorting the above list with Insertion Sort has a time complexity (running time) of $O(n)$
- (c) Sorting the above list with Bubble Sort has a time complexity (running time) of $O(n \log n)$
- (d) Sorting the above list with Selection Sort has a time complexity (running time) of $O(n)$
- (e) Sorting the above list with Bubble Sort has a time complexity (running time) of $O(n)$

Question 40

[2 marks] Below is the code of the Selection Sort algorithm (for sorting a list in ascending (increasing) order) with two lines deleted:

```
def selection_sort(a_list):
    for pass_num in range(len(a_list) - 1, 0, -1):
        pos = 0
        for i in range(1, pass_num+1):
            DELETED_LINE_1
            DELETED_LINE_2
        a_list[pos], a_list[pass_num] = a_list[pass_num], a_list[pos]
```

What code do we need to insert for these two deleted lines in order to get a correctly working Selection Sort algorithm?

- (a) if $a_list[i] < a_list[pass_num]$:
 $pos = pass_num$
- (b) if $a_list[i] > a_list[pass_num]$:
 $pos = pass_num$
- (c) if $a_list[i] > a_list[pos]$:
 $pos = i$
- (d) if $a_list[i] < a_list[pos]$:
 $pos = i$
- (e) None of the other answers.