Question 1:

The worst-time complexity for merge sort is _____.

Answers:

a. O(log n)

b. O(n^2)

c. O(1)

d. O(n log n)

e. O(n)

Question 2:

The best-time complexity for bubble sort is _____.

Answers:

a. O(n^2)

b. O(log n)

c. O(n)

d. O(n log n)

e. O(1)

Question 3:

Fill in the code to complete the following method for sorting a list.

```java-code
public static void sort(double[] list) {

_____;

}

public static void sort(double[] list, int high) {

if (high > 1) {

// Find the largest number and its index
```

```
int indexOfMax = 0;

double max = list[0];

for (int i = 1; i <= high; i++) {

if (list[i] > max) {

max = list[i];

indexOfMax = i;

}

}

// Swap the largest with the last number in the list

list[indexOfMax] = list[high];

list[high] = max;

// Sort the remaining list

sort(list, high - 1);

}

}
```
```

Answers:

a. `sort(list, list.length)`

b. `sort(list)`

c. `sort(list, list.length - 2)`

d. `sort(list, list.length - 1)`

Question 4:

Suppose a list is {22, 29, 25, 24, 28, 21}. After the first pass of bubble sort, the list becomes:

Answers:

a. 22, 25, 29, 24, 28, 21

b. 22, 29, 25, 24, 21, 28

c. 22, 25, 24, 28, 21, 29

d. 22, 29, 25, 24, 28, 21

e. 22, 21, 25, 24, 28, 29

Question 5:

Suppose you choose the first element as a pivot in the list {15, 12, 19, 13, 18, 14, 10, 11, 16, 17}. What is the new list after the partition?

Answers:

a. 14, 12, 13, 10, 11, 15, 16, 17, 19, 18

b. 15, 12, 19, 13, 18, 14, 10, 11, 16, 17

c. 12, 13, 14, 10, 11, 15, 16, 17, 18, 19

d. 14, 12, 11, 13, 10, 15, 18, 19, 16, 17

e. 12, 13, 14, 10, 11, 15, 19, 18, 16, 17

Question 6:

Which one of the following statements is wrong?

Answers:

a. A recursive method can always be replaced by a non-recursive method.

b. Recursive methods usually take more memory space than non-recursive methods.

c. Recursive methods run faster than non-recursive methods.

d. In some cases, however, using recursion enables you to give a natural, straightforward, simple solution

to a program that would otherwise be difficult to solve.

Question 7:

Use the insertion Sort method. Assume the list is {3.1, 3.1, 2.5, 6.4, 2.1}. What is the content of the list

after the first iteration of the outer loop in the method?

Answers:

a. 2.1, 2.5, 3.1, 3.1, 6.4

b. 3.1, 3.1, 2.5, 6.4, 2.1

c. 2.5, 3.1, 3.1, 6.4, 2.1

d. 2.1, 3.1, 2.5, 6.4, 3.1


Question 8:

Suppose we have an arraylist, a\_list and a linkedlist l\_list. Which of the two methods runs faster?

```
// method 1

for(int i = 0; i < n; i++)

    l_list.append(i)


// method 2

for(int i = 0; i < n; i++)

    a_list.append(i)
```

Question 9:

What is the running time of this algorithm

```
int sum = 0;

for (int i =0; i < n; i++) {

for (int j = 0; j < n*n; j++) {

sum += 1

}
```

}

```

Answers:

A O(N)

B O(1)

C O(n^3)

D O(n^2)

Question 10:


Question 11:

Given a sorted singly linked list. What would be the running time of removing duplicates?

Answers:
A O(N)

B O(1)

C O(nlogn)
D O(n^2)

Question 12:

What algorithm/data structure would you use to solve the following problem in O(logN) time.

Given a sorted list of elements and a target element. Find the smallest element in the list larger than the given target.
Answers:
A Hash table
B Binary Search
C Quick Sort
D For Loop


Question 13:

Would Prim's and Kruskals' algorithms for finding the minimum spanning tree still work if the graph had negative weight edges?
Answers:
A False
B True

Question 14:

Given the root of a m-ary tree, root, what algorithm would you use to determine the maximum depth of the tree?
Answers:
A Binary Search
B Depth-First Search
C Breadth-First Search
D None of the above


Question 15:

Given a non-empty list of words, return the k most frequent. How would you solve this in O(nlogn) time. What data structures/algorithms would you use?
Answers:
A Hash Table
B Merge Sort
C Tree
D A and B