11

For growable Array-based Array List implementation compare incremental strategy and the doubling strategy by analyzing the total time T(n) needed to perform a series of 20 push operations. Assume initial array size is 2. How many operations are required for each case. *
(0/10 Puan)

12

In terms of Big-O complexity analysis what is the complexity of the term 3 log n + n + 5 *
(0/5 Puan)

O(logn)

8

Show the results of the following sequence of events, by drawing the state of the data structure: add(2) add(5) add(1) add(7) add(8), add(8), remove(), remove()
Where add and remove are the operations that correspond to the basic operations in a stack *
(5/5 Puan)

(Left is ground, right is top and -> shows next position of stack) 2 -> 2, 5 -> 2, 5, 1 -> 2, 5, 1, 7 -> 2, 5, 1, 7, 8 -> 2, 5, ·

9

The number of operations executed by algorithms A and B is $20*n*logn$ and $2*n^3$, respectively. Determine n0 such that A is better than B for $n \geq n0$. *
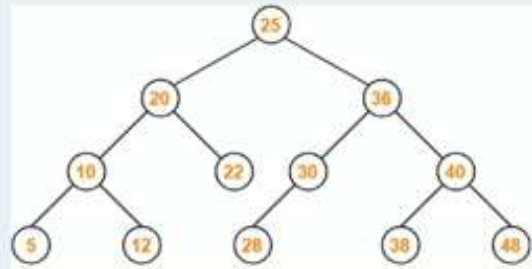(0/10 Puan)

.

10

What is the most important difference between the abstract class and an interface. *
(5/5 Puan)

Extended classes from abstracrt class use "extended", implemented classes from an interface use "implements" and in

6

Show the trace (list the nodes visited) when searching for 34 in the binary seach tree *
(10/10 Puan)

25, 36, 30 then it must be place right of 30

7

Write a simple function with tail recursion *
(10/10 Puan)

```
void print(int n)
{
   if (n < 0)  return -1;
  System.out.println(n);
   print(n-1);
}
```

4

Write a recursive algorithm to compute the sum of all elements in an n×n (two-dimensional) array of integers. What is your running time and space usage? *
(1/15 Puan)

```
public int sum(int[][] data, i, n1, j, n2){
   if (n1 == 1 && n2 == 1) {
      return data[i][j];
   }
   if (n1 == 1) {
      return sum(data, i, n1, j, (n2 / 2)) + sum(data, i, n1, j + (n2 / 2), n2 - (n2 / 2));
   } else {
      return sum(data, i, (n1 / 2), j, n2) + sum(data, i + (n1 / 2), n1 - (n1 / 2), j, n2);
   }
}
```

5

Show the results of the following sequence of events, by drawing the state of the data structure: add(2) add(5) add(1) add(7) add(8), add(8), remove(), remove()
Where add and remove are the operations that correspond to the basic operations in a queue
*

(5/5 Puan)

(left is first, right is last) 2 -> 2, 5 -> 2, 5, 1 -> 2, 5, 1, 7 -> 2, 5, 1, 7, 8 -> 2, 5, 1, 7, 8, 8 -> 5, 1, 7, 8, 8 -> 1, 7, 8, 8

3

What is the complexity/growth rate of the following java function?

```java
public static void printAll(double[ ] x , int a) {
int n = x.length;
for (int j=0; j < a; j++) {
 for (int k=0; k < a; k++) {
    System.out.print(x[j] + x[k]);
   }
 }
} *
```
(0/5 Puan)

1+ (a*a+1) = a^2+2 = O(n)

4

Write a recursive algorithm to compute the sum of all elements in an n×n (two-dimensional) array of integers. What is your running time and space usage? *
(1/15 Puan)

```java
public int sum(int[][] data, i, n1, j, n2){
  if (n1 == 1 && n2 == 1) {
      return data[i][j];
  }
   if (n1 == 1) {
      return sum(data, i, n1, j, (n2 / 2)) + sum(data, i, n1, j + (n2 / 2), n2 - (n2 / 2));
```

For the arithmetic expression 1 – 6 * 2 + 7 < (12 – (4 * 3)) + 8 construct a binary tree and represent it in an array that can be used for the calculation of the result with the postorder traversal. Keep in mind operator ≤ has lower precedence than +/– and * has precedence over +/– . Your answer is just the content of the array (Ex: -12*678) *
(0/10 Puan)

.

Write a recursive method for removing all the elements from a list *
(4/10 Puan)

```
public static void deleteList(Node head) {
    if (head == null)
        return;
    deleteList(head.next);
}
```

Puan: 41/100

1

For the arithmetic expression $1 - 6 * 2 + 7 < (12 - (4 * 3)) + 8$ construct a binary tree and represent it in an array that can be used for the calculation of the result with the postorder traversal. Keep in mind operator $\leq$ has lower precedence than $+/-$ and $*$ has precedence over $+/-$. Your answer is just the content of the array (Ex: -12*678) *
(0/10 Puan)

.

2

Write a recursive method for removing all the elements from a list *
(4/10 Puan)

```
public static void deleteList(Node head) {
    if (head == null)
        return;
    deleteList(head.next);
}
```