Question 1

a) Implement a QUEUE with two STACKs having constant amortized cost for each QUEUE operation (6 points).

Name the two STACKs as Stack1 and Stack2, we can implement the QUEUE as follows:

• ENQUEUE(x): PUSH x into Stack1
• DEQUEUE(x): If Stack2 is not empty, then simply POP from Stack2 and return the element. If Stack2 is empty, POP all the elements of Stack1, PUSH them into Stack2, then POP from Stack2 and return the result.

b) Choose any two from the three methods to prove the amortized cost for each QUEUE operation is O(1) (4 points each).

• Aggregate method
Consider a sequence of n operations. The sequence of operations will involve at most n elements. The cost associated with each element will be at most 4 i.e. (pushed into Stack1, popped from Stack1, pushed to Stack2, and popped from Stack2). Hence, the actual cost of n operations will be upper bounded by T(n) = 4n. Hence, the amortized cost of each operation can be T(n)/n = 4n / n = 4 = O(1).

• Accounting method
We guess that the amortized costs for ENQUEUE and DEQUEUE are 3 and 1. We show that the potential function P(n) satisfies P(n) - P(0) >= 0 for all n.

We have P(0) = 0. If an element is not popped, then it's only pushed twice and popped once. Thus, the cost of 3 is paid for by ENQUEUE operation. The cost for last pop operation is paid for by the DEQUEUE.

Note: Alternatively, we can set the costs for ENQUEUE and DEQUEUE as 4 and 0 respectively.
• Potential method
We guess the potential function \( P(n) = 2 \times \#\text{Elements in Stack}_1 \). \( P(0) = 0 \) and \( P(n) - P(0) \geq 0 \) for all \( n \).

- **ENQUEUE**: Actual cost of PUSH is 1. Number of elements in Stack\(_1\) increases by 1 and Delta \( P \) increases by 2. Amortized cost = actual cost + \( \Delta P = 1 + 2 = 3 \).
- **DEQUEUE**:
  - If Stack\(_2\) is not empty. Actual cost of DEQUEUE is 1. The \#Element in Stack\(_1\) stays the same, i.e. \( \Delta P = 0 \). Amortized cost = actual cost + \( \Delta P = 1 + 0 = 1 \).
  - If Stack\(_2\) is empty. Let \( x = \#\text{Elements in Stack}_1 \). The actual cost of POP is \( 2x \). The \( \Delta P = 0 - 2x = -2x \). Amortized cost = actual cost + \( \Delta P = (2x+1) + (-2x) = 1 \).

Therefore, the amortized costs for ENQUEUE and DEQUEUE are 3 and 1 respectively.

**Question 2**

a) (4 points) Note that the length of a path is the number of nodes along that path. For example, the leftmost path and the rightmost path of the tree below have length 3 and 2, respectively.

```
         1
        / \   /
       9 5 /  \\
      / \ 
     18 16
```

The longest possible length of the rightmost path of a leftist tree of \( n \) nodes will be the largest \( k \) so that \( 2^k \leq n+1 \). When \( n = 16 \), we have \( k = 4 \). The longest possible leftmost path will have length \( n \) when the tree is actually a line. When \( n = 16 \), the longest possible leftmost path has length 16.

If we define the path length as the number of edges along that path, corresponding answers will be 3 for the rightmost path and 15 for the leftmost path.

b) (6 points)

```
   A       B
  / \     / \   
 2   1   9   5
 / \     / \\
6 4   9 5
 / \     / \\
12 10  18 16
 / \     / \\
22 20
```
Merge right tree of B with A

```
     2   5                  1
       / \
      6   4                 9
        / \                / \ 
       12  10             18  16
        / \             / \ 
       22  20           22  20
```

and make it the right tree of B

```
     1
       / \
      9   2
       / \   / \ 
      18  16 6   4
        / \   / \ 
       12  10 5
        / \  
       22  20
```

**Note:** there is NO need to swap the left and right trees in the last step.

**Question 3**

a)  (7 points)

Merge (0, 100, 200, 400) into 700 records.

Merge (700, 600, 700, 900) into 2900 records.

b)  (7 points)

I/O time: \((700 + 2900) / 100 \times 2 \times 2 = 144\) seconds.

CPU time: \((700 + 2900) / 100 \times 1 = 36\) seconds.

Total time: 144 + 36 = 180 seconds.
**Question 4**

a) (4 points) All trees in the binomial heap are binomial trees. Max degree in a binomial tree is $O(\log n)$ (proof by induction).

b) (8 points)

```
37--------10-------------------------------------------- 1
  |       /  \
  41     28   13            6    16   12   25
  |          /  /  \\    /  \
  77             8   14  29  26  23  18
  /     |     |     |     |     |     |
  11  27  38     42
  |     17

37---------10-------------------------------------------- 1
  |       /  \
  41     28   13            6    16   12   25
  |          /  /  \\    /  \
  77             8   14  29  26  23  18
  /     |     |     |     |     |     |
  11  27  38     42

37---------10                      25----12--16------------ 6
  |       /  \
  41     28   13           18   26   23   8   14   29
  |          /  /  \\    /  \
  77             42   11  27  38

25 ------- --12------------------------------------------ 6
  /     |     |     |     |     |     |     |     |
  37  18             10   8   14   29
  /     |     |     |     |     |     |     |
  41     16   28   13   11   17   38
  /     |     |     |     |     |     |
  26  33  77     27
```