Outline

- Question 1. Basic
- Question 2. Basic
- Question 3. Moderate
- Question 4. Moderate
- Question 5. Challenging
Our genius friend, John, has invented the algorithm for sorting the array $A[i..j]$.
**JohnSort(A, i, j)**

Set \( l = j - i + 1 \), \( k = l \% 3 \), \( m = (l - k)/3; \)

if (\( k \neq 0 \)) {
    Find smallest \( k \) items;
    Swap them with items in \( A[i..i+k-1] \) in increasing order;
}

if (\( m == 0 \)) return;

/* Sort remaining 3m items by recursion */
JohnSort(A, i+k, j-m) // Sort first 2m items
JohnSort(A, i + k+m, j) // Sort last 2m items
JohnSort(A, i+k, j-m) // Sort first 2m items
Question 1

(a) Show that the above algorithm is correct.
(b) Give a recurrence for the worst-case running time of JohnSort.
(c) Obtain a tight asymptotic (Θ-notation) bound on the worst-case running time.
   - How does it compare with the worst-case running time of insertion sort and merge sort?
**Question 1**

An example:

The first $2m$ elements ($m=2$)

$K(=2)$

The last $2m$ elements
Question 1
Question 1
Question 1
Question 1

1 3 7 6 2 8 5 4
Question1
Question 1
Question1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question1
Question1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1
Question 1

- In (b), only need to give the recurrence
- In (c), use Master Theorem to bound the time complexity
Question 2

- What’s Radix Sort?
- An example:

```
  2  5  2
  3  3  2
  1  9  4
  2  6  2
```
Question 2
Question 2

\[
\begin{array}{ccc}
3 & 3 & 2 \\
2 & 5 & 2 \\
2 & 6 & 2 \\
1 & 9 & 4 \\
\end{array}
\]
Question 2
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>4</td>
</tr>
</tbody>
</table>
Question 2

1 9 4

2 5 2

2 6 2

3 3 2
Question 2

Illustrate the operation of RadixSort on the following list of English words:

COW, DOG, SEA, RUG, ROW, MOB, BOX, TAB, BAR, EAR, TAR, DIG, BIG, TEA, NOW, FOX

Show clearly how RadixSort works on those data.
Question 3

- Given an array $A[1..k]$ of $k$ strings, with each string representing an integer. Totally there are $n$ digits in these strings.

  Show how to sort them in $O(n)$ time
Question 3

Example:
If the input array is

{ "235", "8", "17", "652", "490", "231562955", "940", "2" },

then $n = 25$
After sorting, we should obtain

\[
\{ \text{"2", "8", "17", "235", "490", "652", "940", "231562955"} \}
\]
Question 3

- A simple **RadixSort** is not capable of solving this problem

- Why?
What's so special about this matrix?
Question 4

Young Tableau is an $m \times n$ matrix with

(i) each row is sorted
(ii) each column is sorted order

Some entries $= \infty$

⇒ Indicate an empty entry

Young Tableau can hold $\leq mn$ finite #s
Is this a Young Tableau?
Question 4

Young Tableau

<table>
<thead>
<tr>
<th>1</th>
<th>4</th>
<th>5</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td><strong>12</strong></td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td><strong>10</strong></td>
<td>16</td>
</tr>
</tbody>
</table>

No!
Question 4

(a) Draw three $4 \times 4$ Young tableaux containing exactly the elements \{9, 16, 3, 2, 4, 8, 5, 14, 12\}.

(b) Given an $m \times n$ Young tableau $Y$

-- Show $Y$ is empty if $Y[1, 1] = \infty$

-- Show $Y$ is full if $Y[m, n] < \infty$
(c) Show how to do Extract-Min on a Young tableau in $O(m + n)$ time

Hint: Extract-Min in a heap

Show that your algorithm is correct.

(d) Show how to use an $n \times n$ Young tableau to sort $n^2$ numbers in $O(n^3)$ time
Young Tableau is similar to a HEAP.

<table>
<thead>
<tr>
<th>1</th>
<th>4</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>
Question 4

Recalling how we do Extract-Min in Heap
Question 4
Question 4

This node may violate the heap structure
Question 4

This node may violate the heap structure.

Observe that each time at most one node may violate the structure.
Question 4

The diagram shows a tree with a root node labeled 9, which has two children: a node labeled 5 and a node labeled 2. The node labeled 5 has three children: nodes labeled 6, 7, and 3, and the node labeled 2 has two children: nodes labeled 4 and 3. The question asks if this tree is a valid 'dochi'.
Question 4

if we choose to swap 9 and 5, another violation occurs.
New violation!
So we choose to swap the minimum among nearby nodes
Question4
Question 4
Question 4
Question 4
Question 4

(d) You need to clearly show why your algorithm’s time complexity is $O(n^3)$
Question 5 (Bonus)

Finding a number in $O(m+n)$ time inside a Young tableau
Question 5 (Bonus)

HINT 1:

This is a boundary that the \( \searrow \) direction contains all elements that is less than 8.
**Question 5 (Bonus)**

**HINT 2:**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>4</th>
<th>5</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>6</td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>10</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

5 is the greatest element in the red square, 12 is the greatest element in the pink square.