In the office hours this week, we discussed the following topics:

- We went over the closed form of \textit{mergeSort} for $\hat{n}$. Recall that the recurrence relation we have for the algorithm is as follows:

$$T(n) = \begin{cases} 1 & n = 1 \\ T\left(\left\lfloor \frac{n}{2} \right\rfloor \right) + T\left(\left\lceil \frac{n}{2} \right\rceil \right) + n + 1 & n > 1 \end{cases}$$

Also, recall that $\hat{n} = 2^k$. Hence,

$$T(\hat{n}) = \begin{cases} 1 & n = 1 \\ 2T\left(\frac{\hat{n}}{2^k}\right) + \hat{n} + 1 & n > 1 \end{cases} \quad (*)$$

We start from the recursive component of $(*)$, and unwind it. We do not simplify it too much in order to see a pattern:

$$T(\hat{n}) = 2T\left(\frac{\hat{n}}{2^k}\right) + \hat{n} + 1$$

unwind it more by plug in $\frac{\hat{n}}{2^k}$ into $T(\hat{n})$ in $(*)$

$$= 2^2 T\left(\frac{\hat{n}}{2^2}\right) + 2^k \hat{n} + 2 + \hat{n} + 1 \quad \text{make sure not to simplify it too much}$$

$$= 2^k T\left(\frac{\hat{n}}{2^k}\right) + 2^{k-1} \hat{n} + 2^{k-1} + \cdots + 2^2 \hat{n} + 2^2 + 2 \hat{n} + 2 + \hat{n} + 1$$

$(*)$ continue unwinding to see a general pattern in terms of $k$

$$= 2^k T\left(\frac{\hat{n}}{2^k}\right) + 2^{k-1} \hat{n} + 2^{k-1} + \cdots + 2^2 \hat{n} + 2^2 + 2 \hat{n} + 2 + \hat{n} + 1$$

since $\hat{n} = 2^k$, $T\left(\frac{\hat{n}}{2^k}\right) = T(1) = 1$, now let’s rearrange the terms

$$= 2^k + 2^{k-1} + \cdots + 2^2 + 2 + \hat{n} + 1$$

$$= 2^{k+1} - 1 + k \cdot \hat{n}$$

$$= 2 \cdot 2^k - 1 + k \cdot \hat{n}$$

$$= 2\hat{n} - 1 + (\lg \hat{n})\hat{n}$$

hence $T(\hat{n}) \in \theta(\hat{n} \lg \hat{n})$
So, we obtained the closed form for the time complexity of \textit{mergeSort} for certain sizes (powers of \(k\)) as: \(T(\hat{n}) = \hat{n} \left(\lg \hat{n}\right) + 2\hat{n} - 1\). Because of “...” in above calculations, one may ask how we know this is correct. So, we should prove it, that I left it as a practice for you.

From the closed form of \(T(\hat{n})\), we concluded that \(T(\hat{n}) \in \Theta(\hat{n} \lg \hat{n})\). (***)

We stated that in the course note, under Lemma 3.6, it has proved that for \textit{mergeSort}:

\[
T\left(\frac{n}{2}\right) \leq T(n) \leq T(\hat{n}) \quad \text{when} \quad \frac{n}{2} \leq n \leq \hat{n} \quad (***)
\]

From (***) we generalized (***) and made the \textit{conjecture} that \(T(n) \in \Theta(n \lg n)\). In the \textit{lecture}, we proved \(T(n) \in \Omega(n \lg n)\), and as a practice I asked you to prove \(T(n) \in O(n \lg n)\).

- Another very important topic that we discussed during the office hours is as follows. Compare how we conducted the asymptotic analysis for \textit{binSearch} last week with the analysis that we did for \textit{mergeSort} this week (mentioned above too).
  - For \textit{binSearch},
    1. We estimated a \textit{rough} closed form for \(T(n)\).
    2. From the rough closed form, we made a conjecture that \(T(n) \in O(\lg n)\).
    3. Then, we proved our conjecture, using induction.
  
  - For \textit{mergeSort},
    1. We obtained an \textit{exact} closed form for \textit{certain} sizes in \(T(n)\), denoted it by \(T(\hat{n})\) where \(\hat{n} = 2^k\). We left the correctness proof of \(T(\hat{n})\) as a practice.
    2. From the exact closed form for \(\hat{n}\) and the lemma that \(T(n)\) is \textit{monotonic} and increasing, i.e. \(T\left(\frac{n}{2}\right) \leq T(n) \leq T(\hat{n})\), we made a conjecture that \(T(n) \in \Theta(n \lg n)\)
    3. Then, we proved our conjecture.

Either of the two approaches above can be applied to the other algorithm (and may apply to any algorithms you see in future).

- We also briefly discussed Q3 of this week’s \textit{tutorial}. In particular, after the following observation

\[
T(n) = T(n - 1) + n - 2 \\
= T(n - 2) + n - 1 - 2 + n - 2 = T(n - 2) + 2n - 5 \\
= T(n - 3) + n - 2 - 2 + 2n - 5 = T(n - 3) + 3n - 9 \\
= T(n - 4) + n - 3 - 2 + 3n - 9 = T(n - 4) + 4n - 14 \\
= T(n - 5) + n - 4 - 2 + 3n - 9 = T(n - 5) + 5n - 20
\]

We made the following table, by which we tried to find a pattern for the last term:

<table>
<thead>
<tr>
<th>(k)</th>
<th>\text{value}</th>
<th>\text{pattern}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>(=0+2)</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>(=2+3)</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>(=5+4)</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td>(=9+5)</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>(=14+6)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[
= \frac{(1*2)}{2-1}+2 \\
= \frac{(2*3)}{2-1}+3 \\
= \frac{(3*4)}{2-1}+4 \\
= \frac{(4*5)}{2-1}+5 \\
= \frac{(5*6)}{2-1}+6
\]
\[ T(n) = T(n-k) + kn - \left(\frac{k(k+1)}{2} - 1 + k + 1\right) \]

The rest should be obvious. We also discussed that too much simplification in unwinding could make it difficult to find the pattern.

- We also discussed common mistakes of Test 1 as follows:
  - Putting quantification inside definition of \( P(n) \) in proof by induction.
  - Putting quantification inside inductive hypothesis in proof by induction.
  - Not assuming arbitrary elements in the inductive hypothesis of structural induction.