Here are your tutorial sections:

<table>
<thead>
<tr>
<th>Surname</th>
<th>Time</th>
<th>Room</th>
<th>TA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-K</td>
<td>Thursday 6-7pm</td>
<td>BA2145</td>
<td>Ethan</td>
</tr>
<tr>
<td>L-R</td>
<td>Thursday 6-7pm</td>
<td>BA2155</td>
<td>Zach</td>
</tr>
<tr>
<td>S-Z</td>
<td>Thursday 6-7pm</td>
<td>BA2175</td>
<td>Ekansh</td>
</tr>
<tr>
<td>A-H</td>
<td>Friday 11am-12pm</td>
<td>UC85</td>
<td>Wen</td>
</tr>
<tr>
<td>I-R</td>
<td>Friday 11am-12pm</td>
<td>UC87</td>
<td>Felipe</td>
</tr>
<tr>
<td>S-Z</td>
<td>Friday 11am-12pm</td>
<td>UC144</td>
<td>Ryan</td>
</tr>
</tbody>
</table>

These exercises are to give you practice applying the Master Theorem to divide-and-conquer algorithms.

Reminder: The Master Theorem can be applied to recurrences of the form:

\[ T(n) = \begin{cases} 
  k & \text{if } n \leq B \\
  a_1 T(\lceil n/b \rceil) + a_2 T(\lfloor n/b \rfloor) + f(n) & \text{if } n > B 
\end{cases} \]

where \( B, k > 0, b > 1, a_1, a_2 \geq 0, \) and \( a = a_1 + a_2 > 0. \) \( f(n) \) is the cost of splitting and recombining.

If \( f \in \theta(n^d) \), then

\[ T(n) \in \begin{cases} 
  \theta(n^d) & \text{if } a < b^d \\
  \theta(n^d \log n) & \text{if } a = b^d \\
  \theta(n^{\log_b a}) & \text{if } a > b^d 
\end{cases} \]

1. A non-empty array \( A \) with integer entries has the property that no odd number occurs at a lower index than an even number. Devise a divide-and-conquer algorithm for finding the highest index of an even number element, or \(-1\) if \( A \) has no elements that are even numbers. Use the Master Theorem to bound the asymptotic time complexity of your algorithm.

2. Consider this informal algorithm for QuickSort of a non-empty array \( A \) of distinct integers

   (a) Choose a pivot, \( p \) from \( A \) in constant time

   (b) Partition \( A \) into \( A_p^- \) consisting of elements less than \( p \), \([p]\) itself, and \( A_p^+ \) consisting of elements greater than \( p \). Recursively QuickSort \( A_p^- \) and \( A_p^+ \)

   (c) Concatenate the sorted version of \( A_p^- \), \([p]\), and the sorted version of \( A_p^+ \)

Write a recurrence \( T \), for the time complexity of QuickSorting \( A \). Assume the worst (that the constant-time choice of a pivot is consistently unlucky), and use repeated substitution to find a closed form for \( T \). Assume the best (that the constant-time choice of a pivot is consistently lucky) and use the Master Theorem to bound \( T \).