Implementing GS Algorithm:

While m is free ① identify a free man.

m proposes to the next woman m on his list. ② Find highest-ranked woman that m hasn't proposed to.

If w is free ③ whether w is free

+ (m, w)

Else if w is engaged to m' ④ w: m > m'?

If w: m > m' then + (m, w) - (m', w)

Else do nothing.

Output all engagements.

Arrays and Linked Lists.

Query: What is the i-th element?

- Whether an element e is in the list? (where is it?)

Update: Add an element to this list.

Remove.

- Arrays

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>7</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Query the i-th element: O(1)

Add/remove an element: O(n) \( n = \text{current length} \)

Check e is in the list or not: O(n)

- Linked Lists

head → 5 ← 7 ← 2 ← 4  \( \text{single linked list} \)

(adding an element)

头 → 7 ← 2 ← 4  \( \text{double linked list} \)

Add/Remove an element: O(1) \( \text{given a pointer to location we want to add} \)

Example: 5 7 2 4  \( \rightarrow 5 7 1 2 4 \)

Check e is in the list: O(n)