Runtime of the GS Algorithm:

- \( \Theta(n^2) \).
  
  Recall: we focus on *worst case* runtime.

  Because # of iterations = \( n^2 \)
  
  All operations take \( O(1) \) in our implementation.

- \( \Theta(n^2) \)?
  
  Yes! But we did not cover it in class.
  
  We did not prove runtime = \( \Omega(n^2) \).
  
  To prove this, we need to construct inputs
  
  with \( n \) men and \( n \) women, such that
  
  our implementation of G-S algorithm takes \( \Omega(n^2) \) time.

- Input size = \( \Theta(n^2) \)
  
  Reading the input takes \( \Theta(n^2) \).

- Even in the model where we can ask
  "Who is \( m_i \)'s \( j \)-th favorite woman?"
  in \( O(1) \) time, G-S algorithm may still
  take \( \Omega(n^2) \) iterations (see Q1 in HW1).

---

Common runtimes:

**Q1**: Find maximum element in an array.

**A1**: \( \max = -\infty \)

  for \( i = 1 \) to \( n \)
  
  if \( A[i] > \max \) then \( \max = A[i] \);
  
  runtime = \( O(n) \)

**Q2**: Decide if there is a consecutive interval
  that sums to \( s \) in the input array.

**Example**: \( A = [1, -6, 2] \)

- \( s = 3 \)
- \( s = -3 \)

  No! \( \quad \) Yes! \( \quad \) \( \text{sum}[1, -6, 2] = -3 \).

**A2**: \( \) for \( i = 1 \) to \( n \)

  \( \) for \( j = i \) to \( n \)

  \( \) \( \sum_{i \leq j} = 0 \)

  \( \) \( \sum_{i \leq j} = \sum_{i \leq j} + A[k] \)

  \( \) if \( \sum_{i \leq j} = s \) return YES.

**O(n^3)**: 3 loops, each loop runs \( O(n) \) time.

**\( \Omega(n^3) \)**: Yes!

  \( i = 1 \ldots \frac{n}{3} \), \( j = \frac{2n}{3} \ldots n \).

  \( \left(\frac{1}{3}n\right)^3 \)

  \( k = i \ldots j \), there are \( \frac{2n}{3} \) choices for \( k \).
Q3: Find a consecutive interval with maximum average.
A3: same as A2, except that we compute
\[ \text{avg}_{i-j} = \frac{\text{sum}_{i-j}}{(j-i+1)} \]
and output the \((i,j)\) with the maximum \(\text{avg}_{i-j}\).
Runtime of A3 = \(O(n^3)\).
This can be solved in \(O(n)\) time.
Observe that maximum-average interval is always a single element.

Q4: Given \(n\) points on 2-D plane, find two points that are closest to each other.

\[
\begin{array}{c}
\text{min}\text{-}\text{dis} = +\infty \\
\text{For } i = 1 \text{ to } n \\
\quad \text{for } j = 1 \text{ to } n \\
\quad \quad \text{if } (i \neq j) \\
\quad \quad \quad \text{min}\text{-}\text{dis} = \min \bigg( \text{min}\text{-}\text{dis}, \sqrt{(x_i-x_j)^2 + (y_i-y_j)^2} \bigg)
\end{array}
\]

\[
\begin{array}{c}
\text{min}(a,b) := \\
\begin{cases} 
\text{if } a < b & \text{return } a \\
\text{else} & \text{return } b
\end{cases}
\end{array}
\]
Runtime of A4: \(O(n^2)\).
This can be done in \(O(n \log n)\).
using a divide-and-conquer algorithm.