Chapter 4, Graphs:

- Concepts and implementation (Friday, Feb 11)
- Traversal (Monday, Feb 14)
- Minimum spanning trees (*Wednesday and Friday*)
- Single-source shortest paths (next week Monday)
- (Start BSTs Wednesday Mar 2)

“Today” (Wednesday and Friday):

- MST problem definition
- Brute-force solution
- General structure of good solutions
- Kruskal’s algorithm, plus proof and analysis
- Prim’s algorithm, plus proof and analysis
- Performance comparison
General strategy for MST (both algorithms):

- Maintain a set of edges $A$ that is a subset of a MST
- At each step, add one edge to $A$ until it’s a MST

Invariant (General MST main loop)

$\text{There exists } T \subseteq E \text{ such that } T \text{ is a minimum spanning tree of } G \text{ and } A \subseteq T.$

General algorithm outline:

$A = \emptyset$
While $A$ isn’t a MST
    add an edge to $A$ that maintains the invariant

Insight 1: $A$ implicitly partitions vertices into connected components. The lightest edge that connects two components is safe.
Lemma (Safe edges in Kruskal’s algorithm.)

If $G = (V, E)$ is a graph, $A$ is a subset of a minimum spanning tree for $G$, and $(u, v)$ is the lightest edge connecting any distinct connected components of $A$, then $(u, v)$ is a safe edge for $A$, that is, $A \cup \{(u, v)\}$ is a subset of a minimum spanning tree.
Proof. Suppose everything in the hypothesis, in particular that $A$ is a subset of some minimum spanning tree $T$ and that $u$ and $v$ are in distinct connected components of $A$, call them $A_u$ and $A_v$. Let $w_T$ be the total weight of $T$, that is, the sum of the weights of all the edges of $T$. We want to prove that adding $(u, v)$ to $A$ makes something that is still a subset of some minimum spanning tree.

If $(u, v) \in T$, then we’re done. Suppose, then, that $T$ does not contain $(u, v)$. Since $T$ is a spanning tree, it means that $u$ and $v$ are connected in $T$. Pick the lightest edge on the path from $u$ to $v$ that is not in $A$, call it $(x, y)$. Essentially $(x, y)$ is an edge that was picked instead of $(u, v)$ that contributed to connecting $A_u$ and $A_v$. 
Snip out \((x, y)\). This would disconnect \(T\), that is, the graph \(T - \{(x, y)\}\) is not a tree, but rather contains two connected components, one with \(u\) in it and the other with \(v\) in it. Now splice in \((u, v)\). That will reconnect \(u\) and \(v\) and make it into a tree again. Formally we’ve made a new spanning tree \((T - \{(x, y)\}) \cup \{(u, v)\}\).

The hypothesis says that \((u, v)\) was the lightest edge connecting distinct components of \(A\). That means \(w(u, v) \leq w(x, y)\). That in turn means that the total weight of the new spanning tree is also just as good, if not better, than the old one: \(w_{T - \{(x, y)\}} \cup \{(u, v)\} \leq w_T\). Since it ties or beats a (supposed) minimum spanning tree, \((T - \{(x, y)\}) \cup \{(u, v)\}\) must be a minimum spanning tree. Therefore \((u, v)\) is safe. □
<table>
<thead>
<tr>
<th></th>
<th>Kruskal</th>
<th>Prim</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unoptimized</td>
<td>Optimized</td>
</tr>
<tr>
<td>Sparse Adjacency list</td>
<td>31579</td>
<td>28841</td>
</tr>
<tr>
<td>Sparse Adjacency matrix</td>
<td>49128</td>
<td>35493</td>
</tr>
<tr>
<td>Medium Adjacency list</td>
<td>147527</td>
<td>54877</td>
</tr>
<tr>
<td>Medium Adjacency matrix</td>
<td>127485</td>
<td>59821</td>
</tr>
<tr>
<td>Dense Adjacency list</td>
<td>136762</td>
<td>69867</td>
</tr>
<tr>
<td>Dense Adjacency matrix</td>
<td>162468</td>
<td>78154</td>
</tr>
</tbody>
</table>