Algorithm Design Process

Hypothesize. How do invariants affect the behavior for each operation?

Identify. What strategies have we used before? What examples can we apply?

Plan. Propose a new way from findings.

Analyze. Does the plan do the job? What are potential problems with the plan?

Create. Implement the plan.

Evaluate. Check implemented plan.

Find a lower and upper bound. Define a slow but totally correct solution. Build a mental model: identify key properties.

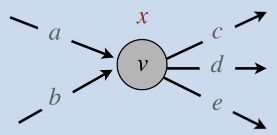
Consider each algorithm that you know. Which ones might work? How do the existing algorithms break down?

Apply an algorithm design idea. Perform a reduction: transform input and output. Or modify the data structures used.

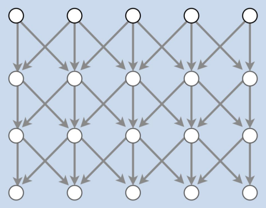
Use an algorithm design paradigm.

Graph Modeling

How to model vertex weights?



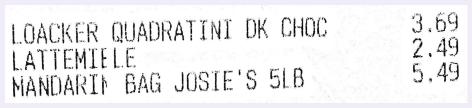
How to model multiple sources and sinks?



Q Line Segmentation

Suppose we're building an **optical character recognition** system.

We want to separate lines of text. There is some white space between the lines but problems like noise and the tilt of the page makes it hard to find.



```
LOACKER QUADRATINI DK CHOC     3.69
LATTEMIELE                     2.49
MANDARIN BAG JOSIE'S 5LB       5.49
```

How can we do line segmentation?

Q Counting Inversions

Given a permutation of length N, count the number of inversions.

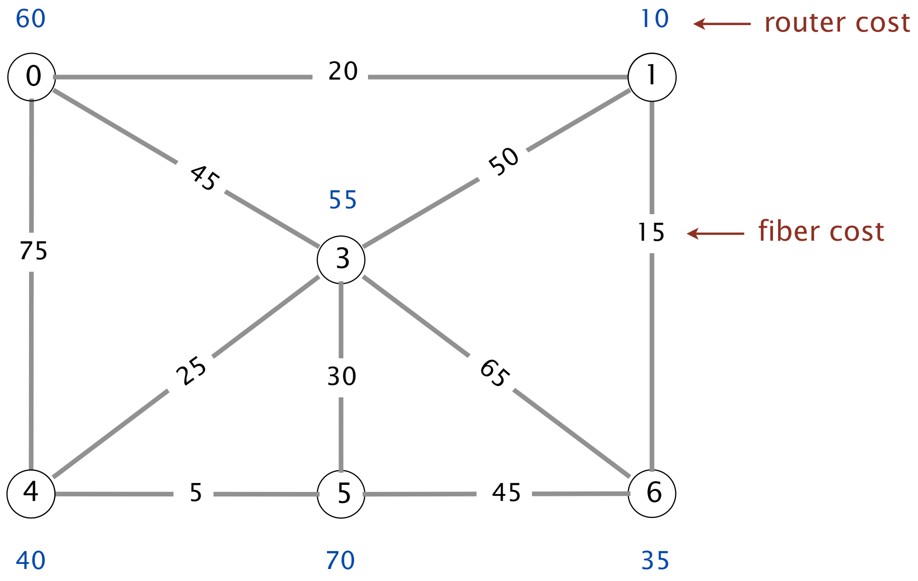
0	**2**	**3**	**1**	**4**	**5**	**7**	**6**

3 inversions: **2–1**, **3–1**, **7–6**

Lower bound? Upper bound? Desired runtime? Algorithm paradigm?

14. **Algorithm design. (8 points)**

There are N dorm rooms, each of which needs a secure internet connection. It costs $w_i > 0$ dollars to install a secure router in dorm room i and it costs $c_{ij} > 0$ dollars to build a secure fiber connection between rooms i and j. A dorm room receives a secure internet connection if either there is a router installed there or there is some path of fiber connections between the dorm room and a dorm room with an installed router. The goal is to determine in which dorm rooms to install the secure routers and which pairs of dorm rooms to connect with fiber so as to minimize the total cost.



This instance contains 6 dorm rooms and 10 possible connections. The optimal solution installs a router in dorm rooms 1 and 4 (for a cost of 10 + 40) and builds the following fiber connections: 0-1, 1-6, 3-4, 4-5 (for a cost of 20 + 15 + 25 + 5).

Formulate the problem as a *minimum spanning tree* problem. To demonstrate your formulation, modify the figure above to show the MST problem that you would solve to find the minimum cost set of routers and fiber connections.

9. **(30 points).** Imagine that we have a list of every commercial airline flight that has ever been taken, stored as an `ArrayList<Flight>`. Each `Flight` object stores a flight start time, a flight ending time, and a number of passengers. These values are all stored as `int`s.

The trick we use to store a flight start time (or end time) as an `int`, rather than as some sort of `Time` object, is to store the number of minutes that had elapsed in the Pacific Time Zone since midnight on January 1st, 1914, which was the first day of commercial air travel.

For example, a flight taking off at 2:02 PM on March 6th, 1917 and landing at 3:03 PM the same day carrying 30 passengers would have takeoff time 1,671,243, landing time 1,671,304, and number of passengers 30.

Give an algorithm for finding the largest number of people that have ever been in flight at once.

Your algorithm must run in N log N time, where N is the number of total commercial flights ever taken. Your algorithm must not have a runtime that is explicitly dependent on the number of minutes since January 1st, 1914, i.e. you can't just consider each minute since that day and count the number of passengers from each minute and return the max.

Your algorithm may use any data structures discussed in the course (e.g. arrays, `ArrayDeque`, `LinkedListDeque`, `ArrayList`, `LinkedList`, `WeightedQuickUnion`, `TreeMap`, `HashMap`, `TreeSet`, `HashSet`, `HeapMinPQ`, etc.)

a. List any data structures needed by your algorithm, including the type stored in the data structure (if applicable). If you use a data structure that requires a `compareTo` or `compare` method, describe **briefly** how the objects are compared. Do not include the provided `ArrayList<Flight>` in your list of data structures. Please list concrete implementations, not abstract data types.

b. Briefly describe your algorithm in plain English. Be as concise and clear as possible.

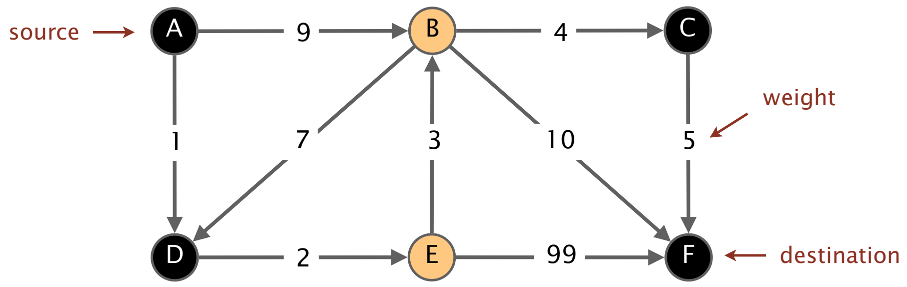
12. **Reductions** (8 points)

Consider the following two graph-processing problems:

- Shortest-Path. Given an edge-weighted digraph G with nonnegative edge weights, a source vertex s, and a destination vertex t, find a shortest path from s to t.

- Shortest-Princeton-Path. Given an edge-weighted digraph G with nonnegative edge weights, a source vertex s, a destination vertex t, and *with each vertex colored black or orange*, find a shortest path from s to t that uses *at most one orange vertex*. Assume that the source vertex is not orange.

In the edge-weighted digraph below, the shortest path from A to F is $A \to D \to E \to B \to C \to F$ (weight 15) but the the shortest Princeton path is $A \to B \to C \to F$ (weight 18).



(a) Give a linear-time reduction from Shortest-Path to Shortest-Princeton-Path. To demonstrate your reduction, draw the edge-weighted digraph (labeling the source and destination vertices and coloring each vertex black or orange) that you would construct to solve the Shortest-Path instance above.