# Today's announcements:

MP7 available. Due 12/8, 11:59p.

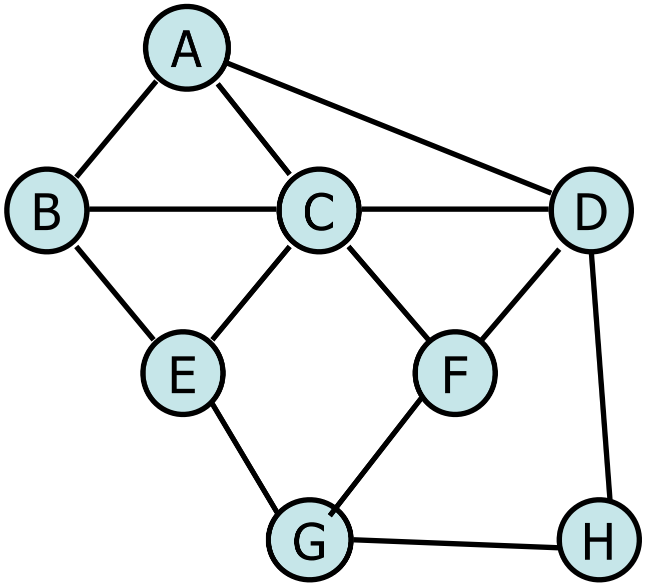
Please check your grade: chara.cs.illinois.edu

Final exam: 12/14, 7-10p, locations TBA

Conflict: email ramais@illinois.edu

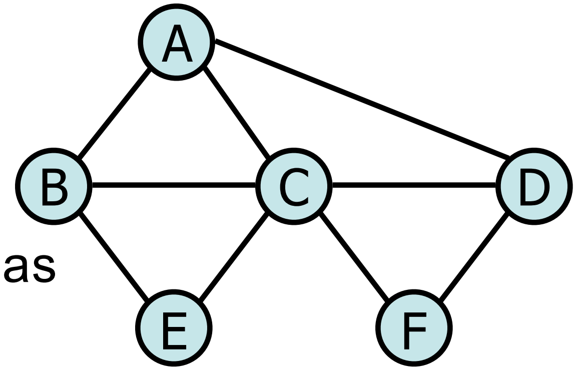
| | |
|---|---|
| A | C B D |
| B | A C E |
| C | B A D E F |
| D | A C F H |
| E | B C G |
| F | C D G |
| G | E F H |
| H | D G |

## Graphs: BFS example



# Graphs: Traversal – BFS

Visits every vertex and classifies each edge as either “discovery” or “cross”

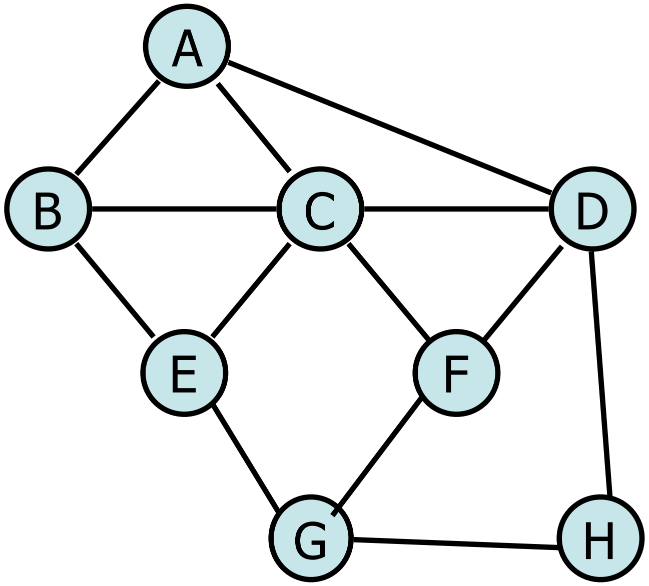


```
Algorithm BFS(G)
    Input: graph G
    Output:  labeling of the edges of G
as discovery edges and back edges


For all u in G.vertices()
  setLabel(u, UNEXPLORED)
For all e in G.edges()
  setLabel(e, UNEXPLORED)
For all v in G.vertices()
  if getLabel(v) = UNEXPLORED
    BFS(G,v)
```

```
Algorithm BFS(G,v)
    Input: graph G and start vertex v
    Output:  labeling of the edges of G in the connected component of v as
discovery edges and cross edges
queue q;
setLabel(v, VISITED)
q.enqueue(v);
While !(q.isEmpty)
  q.dequeue(v)
  For all w in G.adjacentVertices(v)
      if getLabel(w) = UNEXPLORED
          setLabel((v,w),DISCOVERY)
          setLabel(w, VISITED)
          q.enqueue(w)
      else if getLabel((v,w)) = UNEXPLORED
          setLabel((v,w),CROSS)
```

|   |   |
|---|---|
| A | C B D |
| B | A C E |
| C | B A D E F |
| D | A C F H |
| E | B C G |
| F | C D G |
| G | E F H |
| H | D G |

# Graphs: BFS example

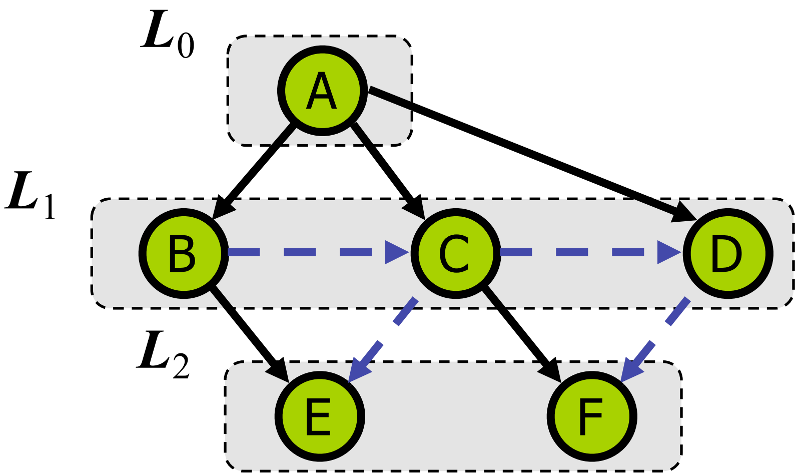


While loop

For loop

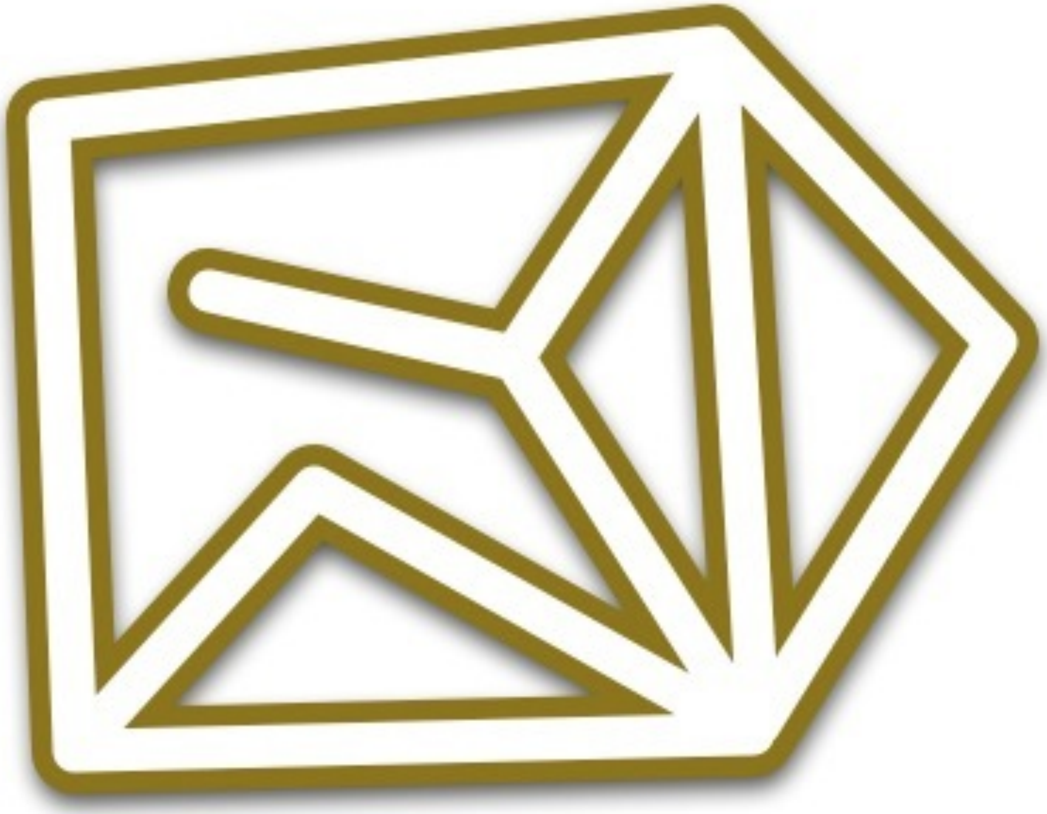
TOTAL RUNNING TIME:

# Graphs: BFS observations



BFS

# Graphs: Traversal - DFS



Ariadne, Theseus, and the Minotaur

http://www.cs.duke.edu/csed/jawaa2/examples/DFS.html

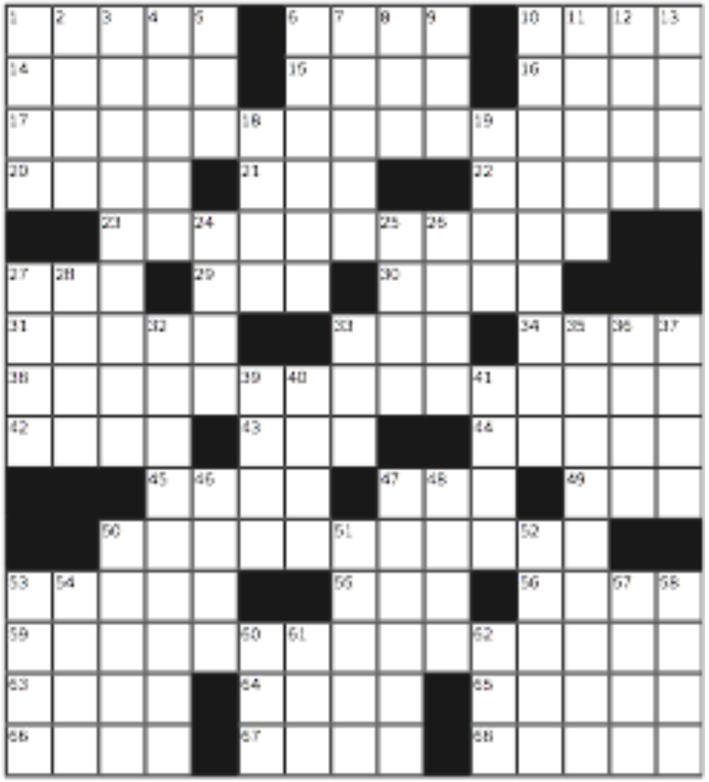
http://www.student.seas.gwu.edu/~idsv/idsv.html

http://www.youtube.com/watch?v=8qrZ1clEp-Y

# Crossword

Edited by Will Shortz

PUZZLE BY AIMEE LUCIDO



11/29/11 (No. 1129)

## ACROSS

1 LPs and 45s
6 Cools, as drinks
10 Traffic components
14 With 5-Down, where "Quiet!" is often yelled
15 "Not guilty," e.g.
16 Eye part
17 Like some stickers
20 Spicy cuisine
21 Sweetie
22 Make fun of
23 Enemy of Spider-Man
27 Identify in a Facebook photo
29 Source of stress for a coll. senior
30 Where shingles go
31 Mea ___
33 Pants part
34 Cutlass or Delta 88
38 Navigation aid for Hansel and Gretel
42 Tale
43 Thumbs-up vote
44 Card game of Spanish origin
45 Almanac contents
47 Not Rep. or Ind.
49 Wood in archery bows
50 Degrees of separation in a Hollywood parlor game
53 Building made of bricks
55 Branch
56 Branch
59 1976 Abba song ... or a hint to the starts of 17-, 23-, 38- and 50-Across
63 Prime draft status
64 Possesses
65 Probably will, after "is"
66 Deborah of "The King and I"
67 Bygone Tunisian V.I.P.'s
68 "Get clean" program

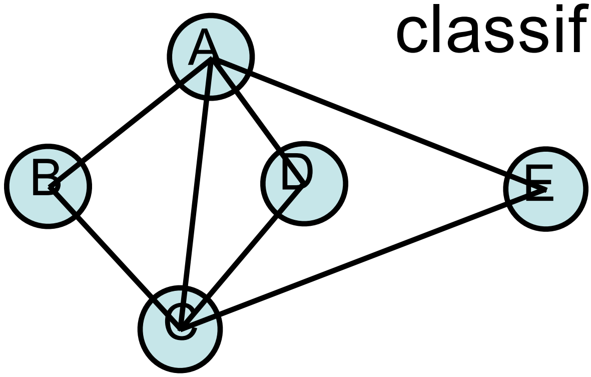
## DOWN

1 "O mighty Caesar! ___ thou lie so low?": Shak.
2 Move slowly
3 One finishing a marathon in eight hours, say
4 Leader of a meeting
5 See 14-Across
6 "There's an app for that" device
7 Trolley sound
8 Day's end, to a poet
9 Downcast
10 Writing with wedges and such
11 Birdlike
12 Ones dressed in stripes, for short
13 Secure
18 Often-impersonated diva
19 Normandy battle site
24 "Gosh almighty!"
25 Utah city
26 Crash and burn
27 Franchise offering "soft serve" and "hand scooped"
28 Subtle glow
32 Chinese zoo attraction
33 Color
35 Feature of the ancient palace of Minos at Knossos
36 Urgent
37 Large amount
39 Ancient Roman censor
40 Actress Meg
41 Tut's resting place
46 ___-deucy (backgammon variety)
47 Compulsion by threat
48 TV award
50 Screwup
51 Mrs. Doubtfire, e.g.
52 Run to Las Vegas, perhaps
53 Crazedly
54 Finished
57 Self-referential, in modern lingo
58 Like many restaurants without a liquor lic.
60 Like some '60s fashion
61 Run a tab
62 Disfigure

## ANSWER TO PREVIOUS PUZZLE

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F | L | O | W | ■ | I | M | A | C | ■ | H | A | D | A | T |
| D | I | D | I | ■ | S | A | R | A | ■ | O | B | E | S | E |
| A | M | O | S | ■ | O | N | A | N | ■ | F | I | C | H | E |
| ■ | P | R | E | T | T | Y | B | O | Y | F | L | O | Y | D |
| ■ | ■ | ■ | G | O | O | ■ | S | E | E | M | E | ■ | ■ | ■ |
| G | A | L | U | M | P | H | ■ | ■ | R | A | N | G | I | N |

DFS: “visits” each vertex
classifies each edge as either “discovery” or “back”

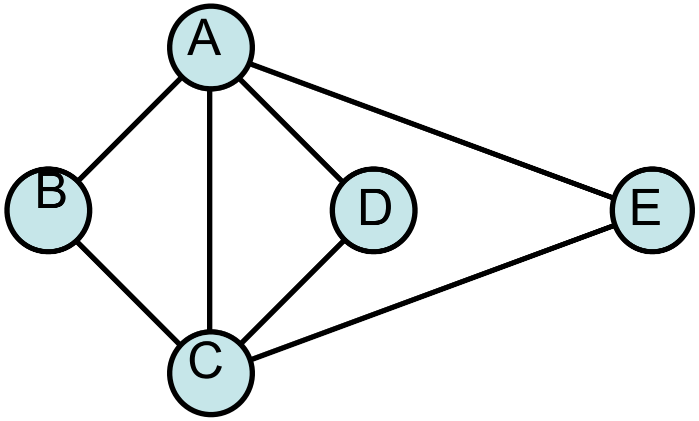


```
Algorithm DFS(G)
        Input: graph G
        Output:  labeling of the edges of G as discovery edges and back edges

For all u in G.vertices()
  setLabel(u, UNVISITED)
For all e in G.edges()
  setLabel(e, UNEXPLORED)
For all v in G.vertices()
  if getLabel(v) = UNVISITED
    DFS(G,v)
```

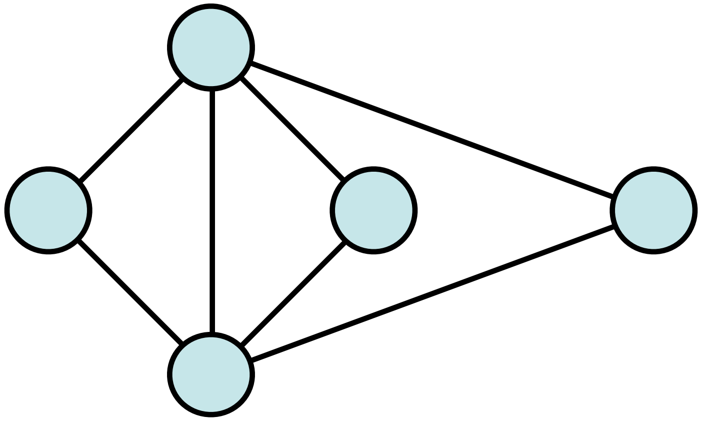
```
Algorithm DFS(G,v)
        Input: graph G and start vertex v
        Output:  labeling of the edges of G in the connected component of v as discovery edges and back edges

setLabel(v, VISITED)
For all w in G.adjacentVertices(v)
    if getLabel(w) = UNVISITED
      setLabel((v,w),DISCOVERY)
      DFS(G,w)
    else if getLabel((v,w)) = UNEXPLORED
      setLabel(e,BACK)
```

| | |
|---|---|
| A | B C D E |
| B | A C |
| C | B A D E |
| D | A C |
| E | A C |

# Graphs: DFS example



# Graphs: DFS Analysis



setting/getting labels

- every vertex labeled twice

- every edge is labeled twice

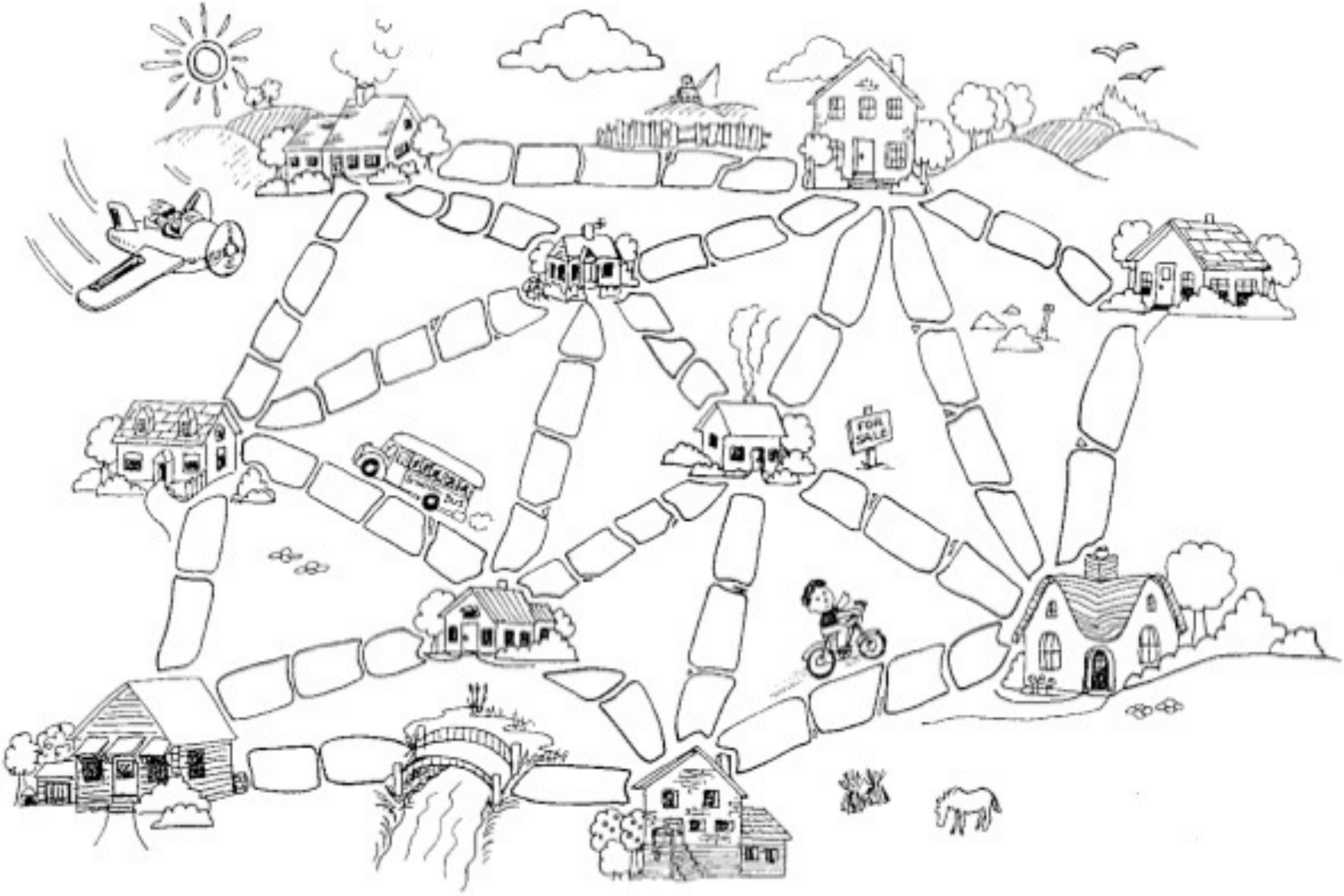
querying vertices

- each vertex
- total over algorithm

querying edges

TOTAL RUNNING TIME:

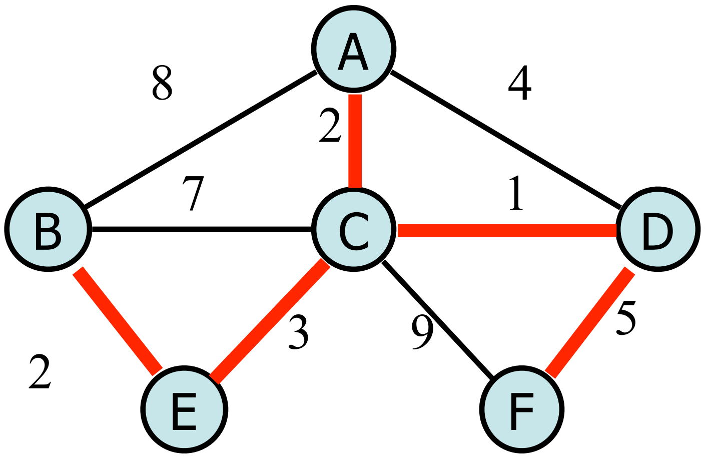
Pause for an example:



# Minimum Spanning Tree Algorithms:

- Input: connected, undirected graph G with unconstrained edge weights
- Output: a graph G' with the following characteristics -
  - G' is a spanning subgraph of G
  - G' is connected and acyclic (a tree)
  - G' has minimal total weight among all such spanning trees -

______________________________


A
B
C
D
E
F
8
4
2
7
1
2
3
9
5