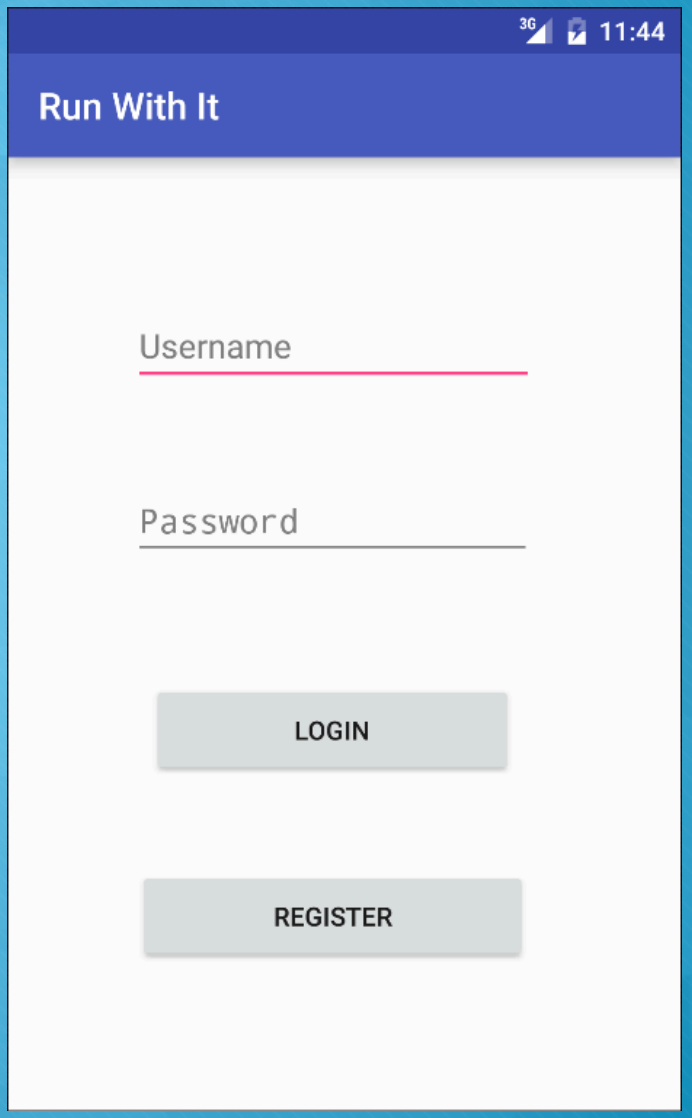
Run With It App Demo

By Gary Zhuo and Jerry Mieles

CS499s1 July 2016

Login Page

- Very simple (as it should be)
- Once users login they'll go straight to the map/route finding page



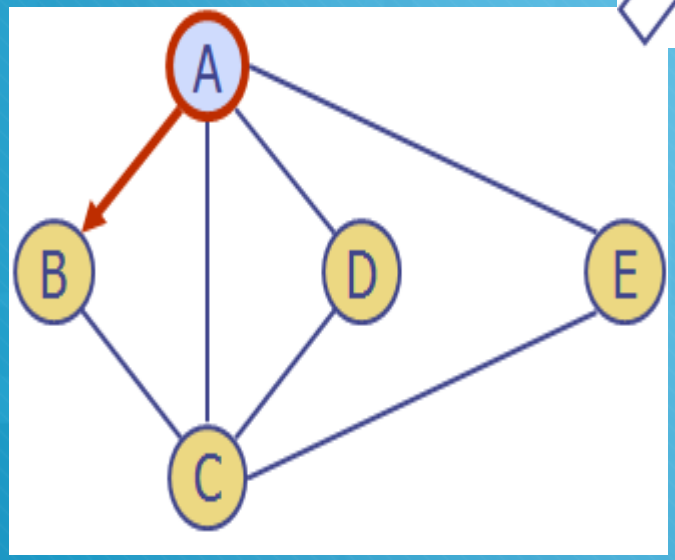
Depth-First Search

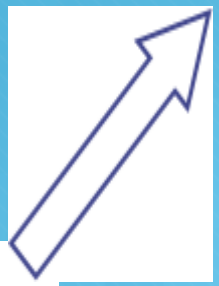
We use DFS as a basic concept for our app algorithm

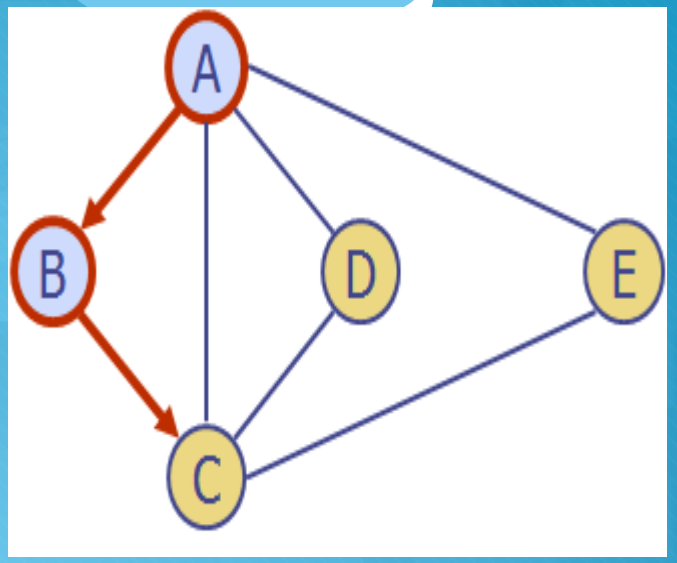
Depth-first search (DFS) is a general technique for traversing a graph

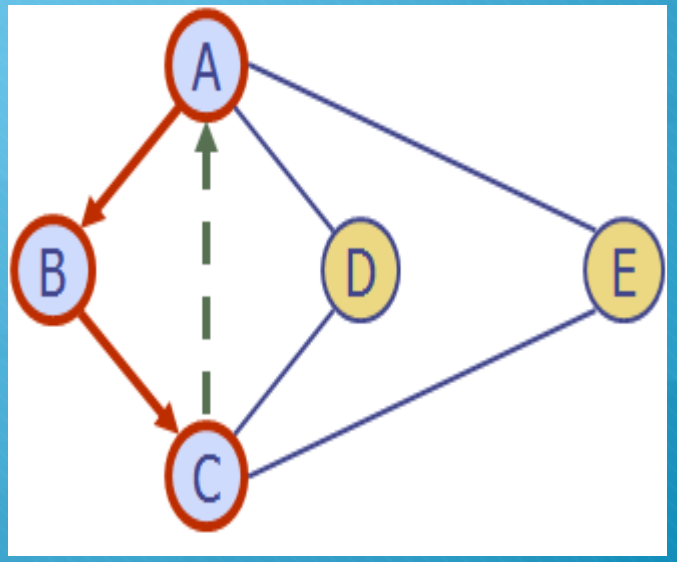
When choosing which vertex to explore next, we favour children over siblings

Example

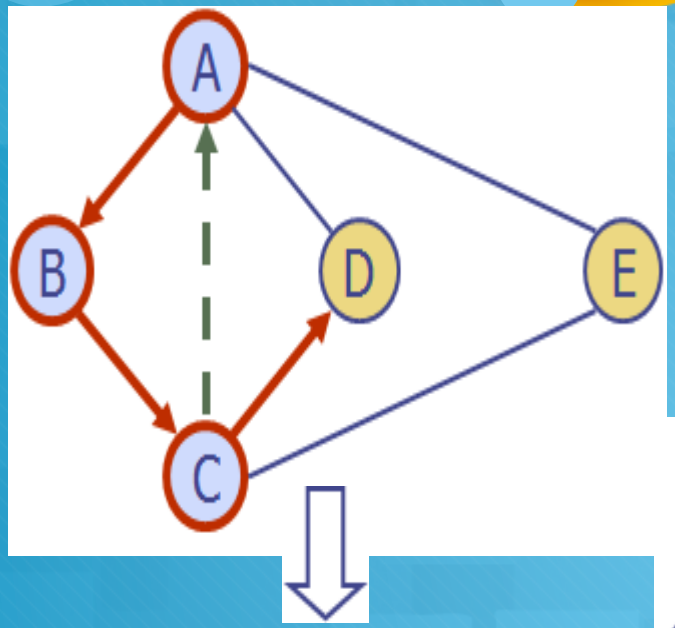


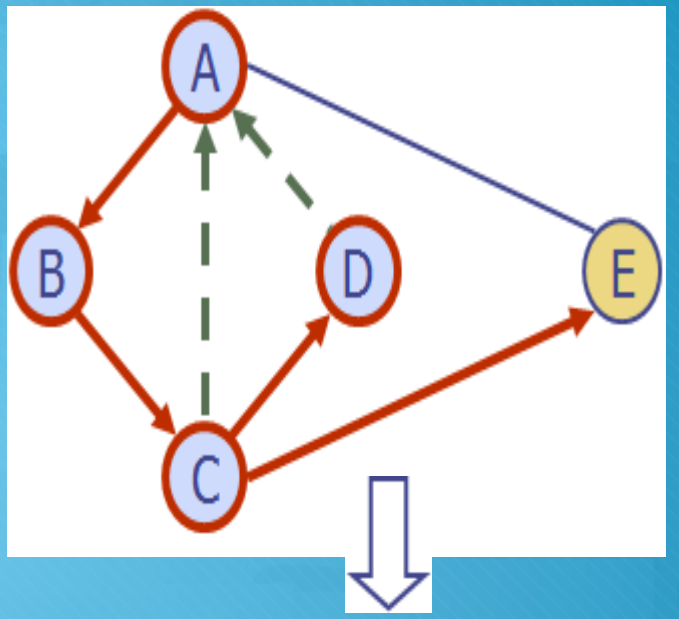


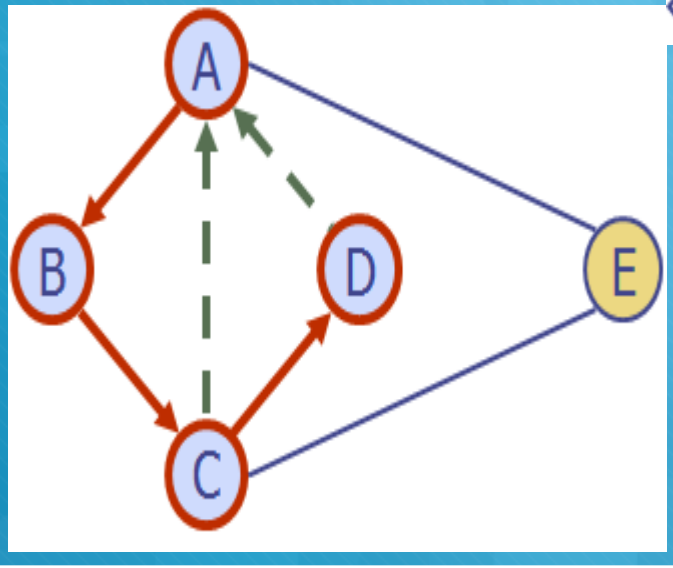


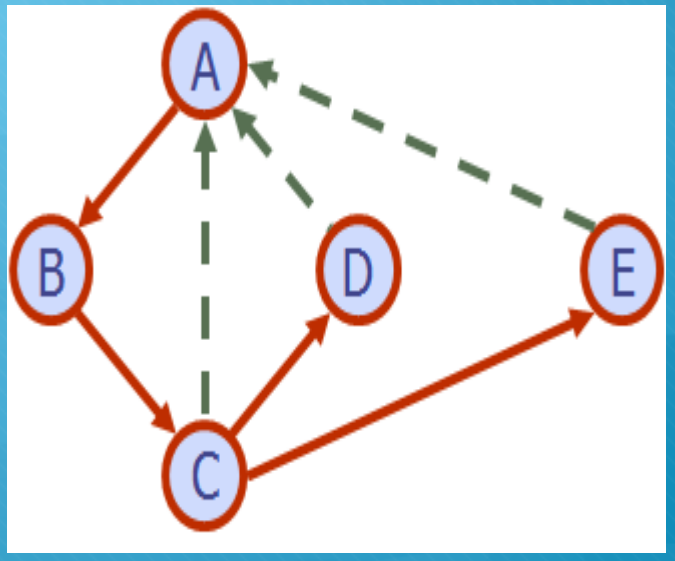


Example (cont.)









Path Finding

the DFS algorithm can be specialized to find all paths between two given vertices.

- It always choose a adjacent vertex as the next one
- Everytime it goes back, it find a possible route

Our algorithm

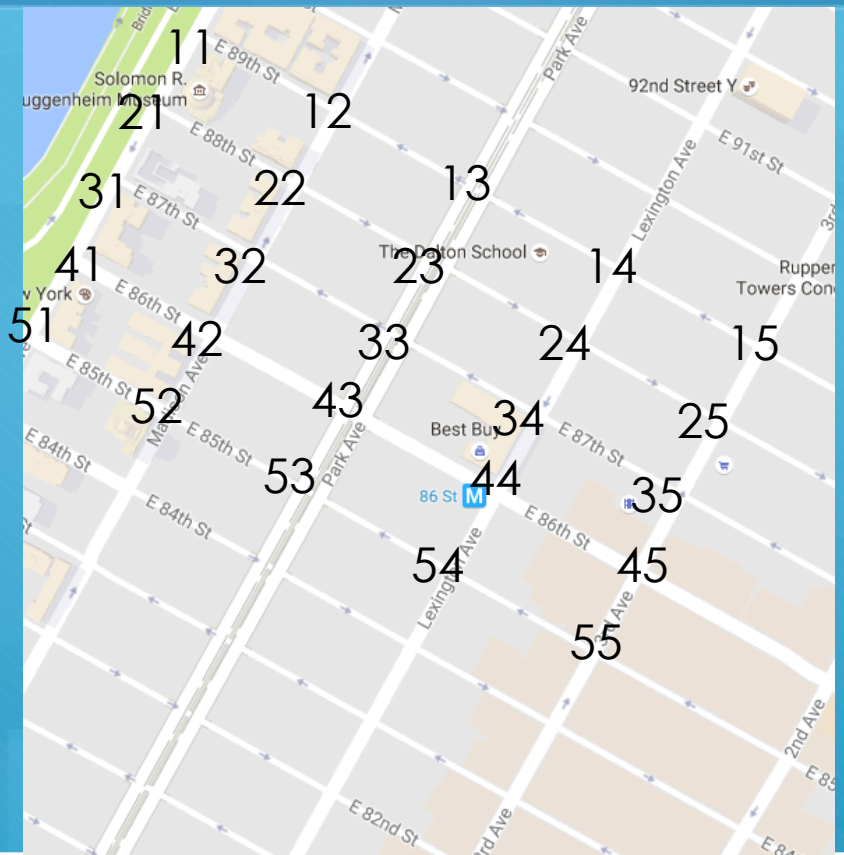
```
DFS(depth,v,path):
path.push(v); //add the current vertex to the path
visited.push(v);
if (depth == 0 && v is target):
print path
 Return //stop clause for successful branch
if (depth == 0):
  return  //stop clause for non successful branch return
for each vertex u such that (v,u) is an edge:
      DFS(depth-distance(v,u),u,path) //recursively check all paths for
of shorter depth
        path.removeLast(); // clean up environment
        visited.removeLast();
```

Our algorithm(cont)

- How to find cycles?
  We find all routes ending in the vertices which are adjacent to the start vertex.
  Then add the start vertex in the end to be a cycle.
- Gain all cycles with specific length after DFS
- Difficulty of routes: sum of all absolute values of difference of elevations

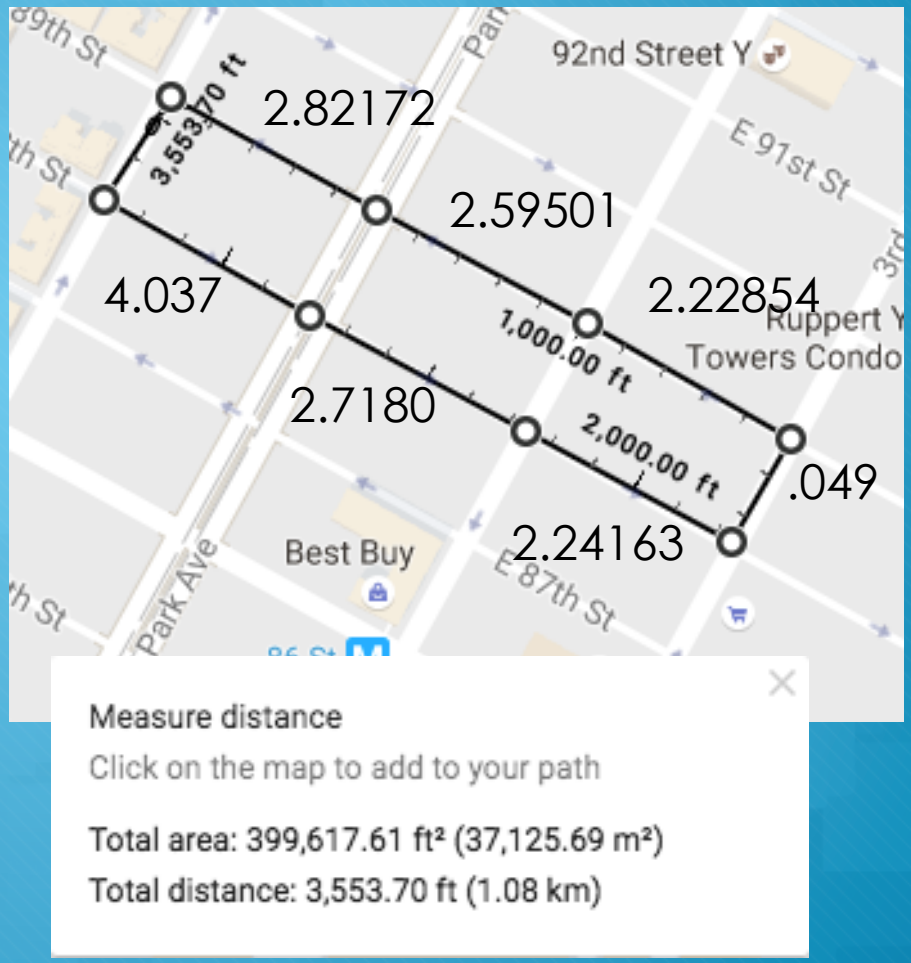
Selected routes match users' preference!

Database grid



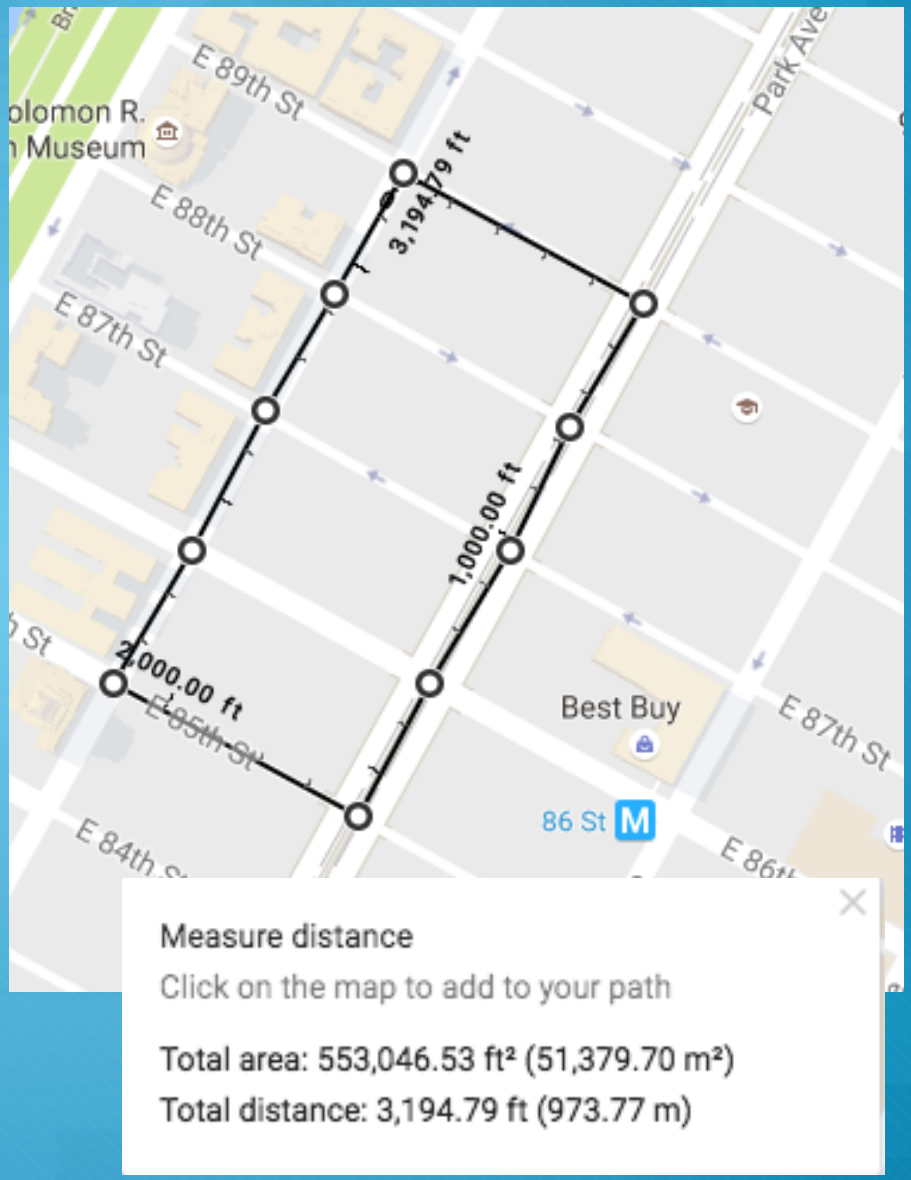
Route 4

Goes through points 12, 13, 14, 15, 25, 24, 23, 22



Route 6

Goes through points 12, 13, 23, 33, 43, 53, 52, 42, 32, 22, 21



Route 10

Goes through points 12, 22, 32, 42, 43, 33, 23, 24, 14, 13