Lecture Title: Breadth-First Search Algorithm.

* Breadth First Search Algorithm

Initialize: open=[start]; closed=[ ]; parent[start]=null; found=no;
While open<>[ ] do
  Begin
    - Remove the first state from left of open, call it X;
    - If X is a goal then {found=yes; break}
    - Generate all possible children of x and put them in list L;
    - Put X on closed;
    - Eliminate from L any child already on closed;
    - Eliminate from L any child in open;
    - For each child y in L set parent[y]=X;
    - Add L to the right of open;
  End.
If found =yes then
  Generate and return the solution path.
Else
  Output no solution.
**Example:** use breadth first search to find the path between A and L for the following search space

**Solution:**
Start=[A], goal=L.

**Iteration #0:**
Open=[A], closed=[ ], parent[A]=null.

**Iteration #1:**
X=A, L=[B,C,D,E], open=[B,C,D,E], closed=[A].

**Iteration #2:**
X=B, L=[F,C], open=[C,D,E,F], closed=[A,B].
Parent[F]=B.

Iteration #3:
X=C, L=[F,G,H], open=[D,E,F,G,H], closed=[A,B,C].

Iteration #4:
X=D, L=[H], open=[E,F,G,H], closed=[A,B,C,D].

Iteration #5:

Iteration #6:

Iteration #7:

Iteration #8:

Iteration #9:

Iteration #10:

Iteration #11:
Parent[N]=I.

Iteration #12:

Iteration #13:
Iteration #14:
\[ X = L \]
Since L is a goal, stop and find path.

Path: \[ A \rightarrow C \rightarrow X \rightarrow L \]