Objectives:

1. Introduce Breadth-first search and Depth-first search.

Reading Assignment:

Neapolitan and Naimipour: Chapters 5-6.

Contents:

1. Outline Depth-first Search Algorithm. (1/2 hour)
2. Breadth-first Search with Branch-and-Bound pruning. (1/2 hour)
3. Best-first Search. (3/4 hour)
4. Describe Depth-first Search. (1/2 hour)
5. Outline Depth-first Search Algorithm. (1/2 hour)
1. Outline Breadth-first Search Algorithm. 

```
Void breadth_first_tree_search(tree T)
{
    queue_of_node Q;
    node u, v;

    initialize(Q);
    v = root of T;
    visit v;
    enqueue(Q, v);
    while(!empty(Q))
    {
        dequeue(Q, v);
        for(each child u of v)
        {
            visit u;
            enqueue(Q, u);
        }
    }
}
```

2. Breadth-first Search with Branch-and-Bound pruning.  
- Illustrate using the 0-1 Knapsack Problem. 

```
Void best_first_branch_and_bound(state_space_tree T, number& best)
{
    priority_queue_of_node PQ;
    node u, v;
    initialize(PQ);
    v = root of T;
    best = value(v);
    insert(PQ, v);
    while(!empty(PQ))
    {
        remove(PQ, v);
        if (bound(v) is better than best)
        {
            for(each child u of v)
            {
                if (value(u) is better than best)
                {
                    if (bound(u) is better than best)
                    {
                        best = value(u);
                        if (bound(u) is better than best)
                        {
                            insert(PQ, u);
                        }
                    }
                }
            }
        }
    }
```
4. Describe Depth-first Search. (1/2 hour)
   - backtracking
   - pre-order tree

5. Outline Depth-first Search Algorithm. (1/2 hour)
   Void depth_first_tree_search(node v)
   {
     node u;
     visit v;
     for(each child u of v)
       depth_first_tree_search(u);
   

Handouts, etc. for Lecture: None.