Trees
Recursive data structures

Trees are everywhere

One of these is not a tree

Binary trees
- A binary tree has at most two branches at each node
- Top node of a binary tree is the root node
- Nodes with no branches are leaves
- Subtrees below a node are the left and right subtrees
- Nodes store values
- We will represent a binary tree using a cell array of 3 elements:
  (value (left subtree) (right subtree))
A binary forest

```matlab
>> seed = {'one' {} {}};
    one

>> twig = {'one' {'two' {} {}} {'three' {} {}}};
    one
    two
    three

>> elm = {'one' {'two' {} {'four' {} {}}} {'three' {'five' {} {}} {}}};
```

Phylogenetic tree of trees

```matlab
phylo = {'trees' {'deciduous' {'elm' {} {}} {'maple' {'japanese' {} {}} {'sugar' {} {}}}} {'evergreen' {'spruce' {} {}} {'pine' {'white' {} {}} {'southern yellow' {} {}}}}};
```

Processing trees

Trees are defined recursively so we will process them recursively.

To solve a problem recursively:
If it is simple enough then solve it
Otherwise express solution in terms of smaller, similar problem(s)

Print the contents of a tree

```matlab
phylo = {'trees' {'deciduous' {'elm' {} {}} {'maple' {'japanese' {} {}} {'sugar' {} {}}}} {'evergreen' {'spruce' {} {}} {'pine' {'white' {} {}} {'southern yellow' {} {}}}}};
```
Think recursively!
If the tree is empty
   do nothing
Otherwise
   print the string in the root node
   print the contents of the left subtree
   print the contents of the right subtree

Let's write the code
If the tree is empty
   do nothing
Otherwise
   print the string in the root node
   print the contents of the left subtree
   print the contents of the right subtree

function printTree (tree)
% prints all of the strings contained in the input tree
if ~isEmpty(tree) % if tree is not empty…
   disp(tree{1}); % print string in root node
   printTree(tree{2}); % print left subtree
   printTree(tree{3}); % print right subtree
end

So is it empty?
function answer = isEmpty (tree)
% returns true if the input tree is an empty cell array
   answer = (length(tree) == 0);
Count the nodes in a tree

If the tree is empty
  return zero
Otherwise
  count nodes in left subtree
  count nodes in right subtree
return sum of counts plus one

Count the nodes in a tree

If the tree is empty
  return zero
Otherwise
  count nodes in left subtree
  count nodes in right subtree
return sum of counts plus one

function nodes = countNodes(tree)
  if isEmpty(tree)
    nodes = 0;
  else
    nodes = 1 + countNodes(tree{2}) + countNodes(tree{3});
  end

Exercise: Modify countNodes to compute the height of a tree

Search for a string in a tree

>> found = searchTree(phylo, 'pine')
pine
white
southern yellow
found =
1
>> found = searchTree(phylo, 'stella')
found =
0

Write searchTree

Let's play twenty questions

>> twentyQuests(twenty)
four legs? yes
hooves? no
stripes? yes
tiger? yes
Wow! I won!
Play again? yes
four legs? no
fly? yes
feathers? no
butterfly? no
Oh well, better luck next time
Play again? no
>>
Game tree for twenty questions

```
 Twenty = {'four legs? {'hooves? {'long neck? {'giraffe? {} {}}}} ...
           {'stripes? {'tiger? {'fleas? {} {}}}} ...
           {'fly? {'feathers? {'eagle? {} {}}} ...
            {'swim? {'penguin? {} {}}} ...
            {'awal? {'penguin? {} {}}} }};
```

TwentyQuests

```
function twentyQuests (questTree)
% plays the twenty questions game, using a binary tree that
% contains questions in the nodes
playAgain = 1;
while playAgain % keeps going as long as user answers yes to
    playAgain = strcmp(input('Play again? ', 's'), 'yes');
    twentyQuests(questTree);
end
```

TwentyQuests

```
function twentyQ (questTree)
% recursive function that implements the twenty questions game
reply = input(questTree{1}, 's');  % print question in root node and
if isLeaf(questTree)                % input user's reply (yes or no)
    if strcmp(reply, 'yes')        % leaf node is reached, so
        disp('Wow! I won!');        % determine outcome of game
    else
        disp('Oh well, better luck next time');
    end
else % leaf node not yet reached
    if strcmp(reply, 'yes')                % if user replied yes, follow left subtree
        twentyQ(questTree{2})
    else  % otherwise follow right subtree
        twentyQ(questTree{3})
    end
end
```