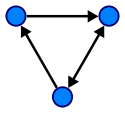
**Graphs Data Structures**

Graphs evolved from the field of mathematics. They are primarily used to describe a model that shows the route from one location to another location.

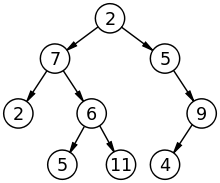
A graph consists of a set of nodes and a set of edges. An edge is a pair of nodes that are connected. A path is the term used to describe traveling between nodes that share an edge. The image below shows a graph with 3 nods and 3 edges.

****

**Tree Data Structure**

A tree data structure, like a graph, is a collection of nodes. There is a root node. The node can then have children nodes. The children nodes can have their own children nodes called grandchildren nodes.

This repeats until all data is represented in the tree data structure. The image below shows a tree data structure.



A tree is a graph that has no cycles (a cycle being a path in the graph that starts and ends at the same vertex). A child node can only have one parent. For this reason trees are not a recursive data structure.

**Why Use Graphs and Trees as Data Structures?**

In computer programming, trees are used all the time to define data structures. They are also used as the basis for algorithms to solve problems.

The most common implementations of a graph are finding a path between two nodes, finding the shortest path from one node to another and finding the shortest path that visits all nodes.

The traveling salesman problem is a great example of using a tree algorithm to solve a problem.

Class Activity

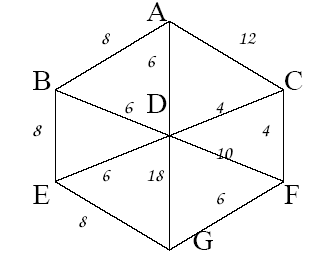
Q1. A Robot is in cell A, wants to go to cell I, order of movement is (up, down, right, left) and blocks B, C and H are blocked. Draw the path (tree) for:

1. Depth first search

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

1. Breadth first search
2. Best first search
3. A\* search

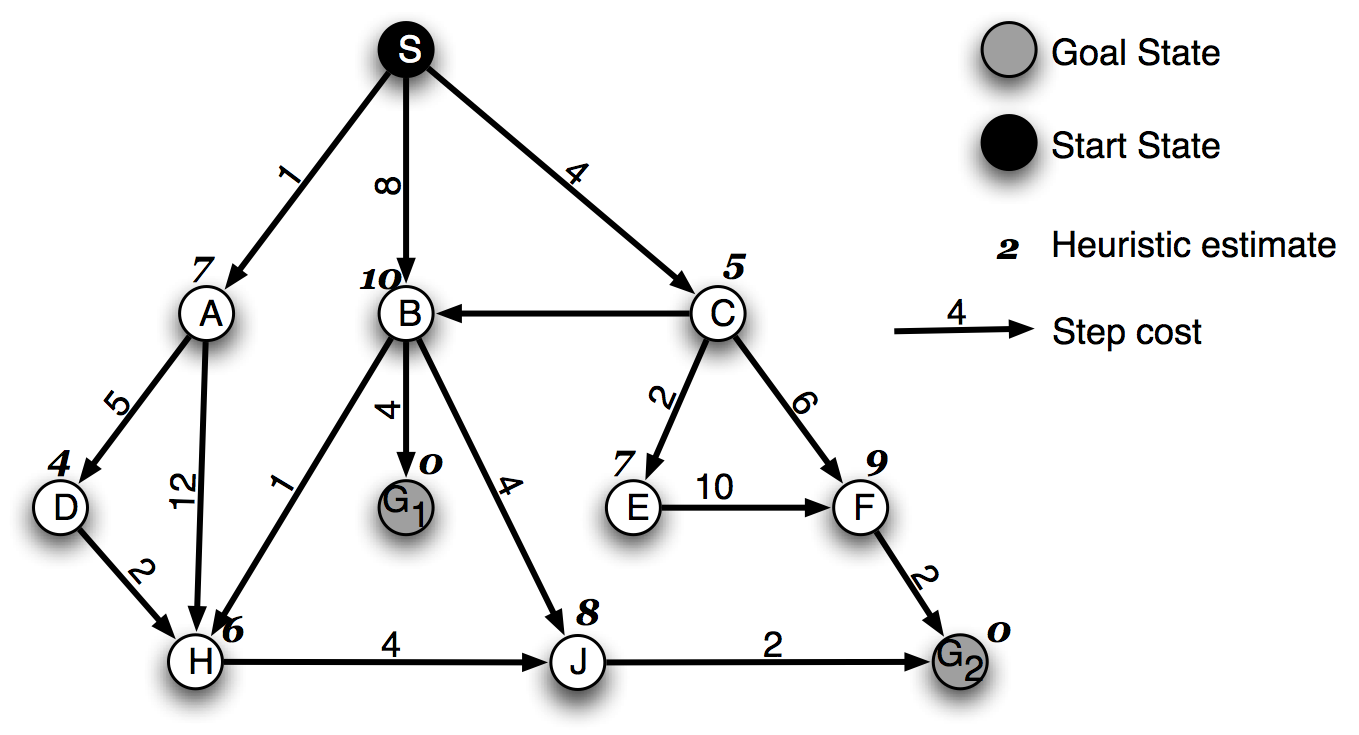
Q2. This question is related with Algorithm A\*:

The numbers in the graph below are the real distances between the nodes. The estimated distances to the goal node G are the following: A: 18, B: 13, C: 8, D: 18, E: 5, F: 6. (A is the start node) 

Do a search with algorithm A\*. You can either show the successive open and closed list marked with the merit values for the nodes, or you can show the search tree, also with the values. Mark also in the tree the order in which the nodes are expanded.

Q3. Consider the search space below, where S is the start node and G1 and G2 are goal nodes. Arcs are labeled with the value of a *cost function*; the number gives the cost of traversing the arc. Above each node is the value of a *heuristic function*; the number gives the estimate of the distance to the goal. Assume that uninformed search algorithms always choose the left branch first when there is a choice. Assume that the algorithms do *not* keep track of and recognize repeated states. For each of the following search strategies, (a) Indicate which goal state is reached first (if any) and

(b) list *in order,* all the states that are popped off the OPEN (un-explored) list. (40)



**3**

***SOLUTION***

Depth-first

(a) -----------

(b) -----------

Iterative Deepening

(a) -----------

(b) -----------

Breadth-first

(a) ------------

(b) ------------

A\* search

(a) ------------

(b) ------------

Q4. Solve by A\* Search Algorithm: Show the complete steps along with the tree.

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 8 |  | 4 |
| 7 | 6 | 5 |

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
| 1 | 6 | 4 |
| 7 |  | 5 |

🡪