LAB # 06

ImplementING informed search techniquies

# OBJECTIVE

Finding path using Greedy best first search.

# THEORY

Greedy best first search is an informed search, which expands the node that appears to be closest to goal. Greedy Best First Search tries to expand the node that is closest to the goal, on the grounds that this is likely to lead to a solution quickly. Thus, it evaluates nodes by using just the heuristic function; that is,

**f(n) = h(n).**

h(n), the cost to get from the node to the goal: f(n) = g(n) + h(n) . Where f(n) = estimated cost of the cheapest solution through n.

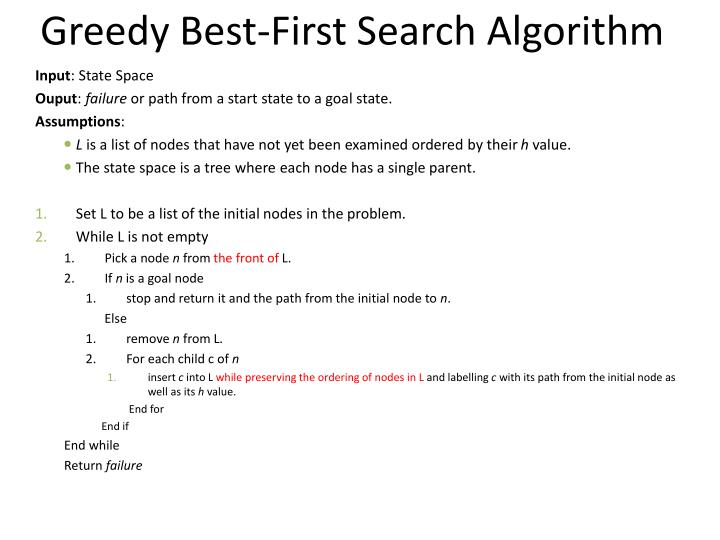
**Properties of greedy best-first search:**

* Complete? No – can get stuck in loops
* Optimal? No
* Time? but a good heuristic can give dramatic improvement
* Space? keeps all nodes in memory

**Simple Greedy best first search Algorithm:**

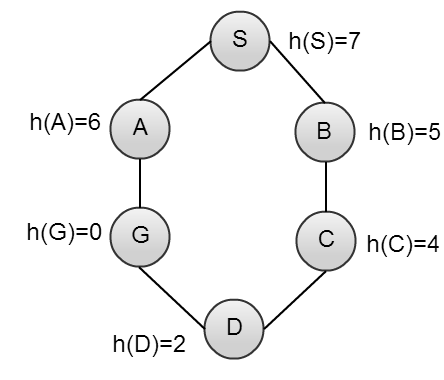
Using a [greedy algorithm](https://en.wikipedia.org/wiki/Greedy_algorithm), expand the first successor of the parent. After a successor is generated:

* If the successor's heuristic is better than its parent, the successor is set at the front of the queue (with the parent reinserted directly behind it), and the loop restarts.
* Else, the successor is inserted into the queue (in a location determined by its heuristic value). The procedure will evaluate the remaining successors (if any) of the parent.



**EXERCISE:**

Consider the following graph. If there is ever a decision between multiple neighbor nodes in the greedy best first search, find the path between S to D. Represent a graph with heuristic using list or dictionaries. The keys of the dictionary represent nodes; the values have a list of heuristic.



Define function name ‘basic\_greedy’, this function keep track of all the visited nodes with greedy best first, is as simple as implementing the steps of the algorithm and assign ‘queue’ variable  already has a node to be checked, i.e., the starting vertex that is used as an entry point to explore the graph.