Name : Priyank Devpura **UIN:225002461**

Programming Assignment I

User Manual:

The programming is done using python

C**ommand to run the code:**

game.py <game_algorithm> <inputs>

game_algorithm: min-max or alpha-beta

inputs: 1,2,3,4,5 (refer nodelist table just below corresponding to these numbers)

For example, lets take nodelist corresponding to input 2 to run alpha-beta.

Cmd: game.py alpha-beta 2

These input numbers are labels which have dictionary that picks the list of nodes based on the input numbers. Following is the look up table:

Inputs	Node list
1.	((4, (7, 9, 8), 8), (((3, 6, 4), 2, 6), ((9, 2, 9), 4, 7, (6, 4, 5))))
2.	(((1, 4), (3, (5, 2, 8, 0), 7, (5, 7, 1)), (8, 3)), (((3, 6, 4), 2, (9, 3, 0)), ((8, 1, 9), 8, (3, 4))))
3.	(5, (((4, 7, -2), 7), 6))
4.	((8, (7, 9, 8), 4), (((3, 6, 4), 2, 1), ((6, 2, 9), 4, 7, (6, 4, 5))))
5.	(((1, (4, 7)), (3, ((5, 2), (2, 8, 9), 0, -2), 7, (5, 7, 1)), (8, 3)), (((8, (9, 3 ,2), 5), 2,(9, (3, 2), 0)), ((3, 1, 9), 8, (3, 4))))

Results for Min-max

Inputs	Output Path	Max Value
1.	[2, 1, 3]	6
2.	[1, 1, 2]	4
3.	[2, 2]	6
4.	[1, 3]	4
5.	[2, 1, 1, 3]	5

Results of alpha-beta

	Output Path	Cuts	Max Val
1.	[2, 1, 3]	MAX CUT after 7 in subtree (7, 9, 8) MIN CUT after 3 in subtree (3, 6, 4) MIN CUT after 2 in subtree (9, 2, 9) MAX CUT after 7 in subtree ((9, 2, 9), 4, 7, (6, 4, 5))	6
2.	[1, 1, 2]	MIN CUT after 2 in subtree (5, 2, 8, 0) MAX CUT after 7 in subtree (3, (5, 2, 8, 0), 7, (5, 7, 1)) MAX CUT after 8 in subtree (8, 3) MIN CUT after 3 in subtree (3, 6, 4) MIN CUT after 3 in subtree (9, 3, 0) MIN CUT after ((3, 6, 4), 2, (9, 3, 0)) in subtree (((3, 6, 4), 2, (9, 3, 0)), ((8, 1, 9), 8, (3, 4)))	4
3.	[2, 2]	MIN CUT after 4 in subtree (4, 7, -2)	6
4.	[1, 3]	MAX CUT after 9 in subtree (7, 9, 8) MIN CUT after 3 in subtree (3, 6, 4) MIN CUT after ((3, 6, 4), 2, 1) in subtree (((3, 6, 4), 2, 1), ((6, 2, 9), 4, 7, (6, 4, 5)))	4
5.	[2, 1, 1, 3]	MAX CUT after 5 in subtree (5, 2) MAX CUT after 8 in subtree (2, 8, 9) MIN CUT after 0 in subtree ((5, 2), (2, 8, 9), 0, -2) MAX CUT after 7 in subtree (3, ((5, 2), (2, 8, 9), 0, -2), 7, (5, 7, 1)) MAX CUT after 8 in subtree (8, 3) MAX CUT after 9 in subtree (9, 3, 2) MIN CUT after (3, 2) in subtree (9, (3, 2), 0) MIN CUT after 3 in subtree (3, 1, 9) MAX CUT after 8 in subtree ((3, 1, 9), 8, (3, 4))	5

Analysis:

The minmax algorithm gives a complete solution and is optimal if the adversary also plays in an optimal way. The min-max works similar to Depth first search except at each level we consider max or min of utility values at that level. Since underneath min-max algorithm, it uses DFS whose time complexity is O(b^d) which is exponential and space complexity is O(bd) where b is branching factor and d is depth.

To improve the time complexity, we apply alpha-beta pruning where we prune a subtree if that subtree does not affect the parent output and return the suboptimal value. With the perfect ordering of utility values in alpha beta pruning, we can achieve time complexity of O(b^(d/2)) . Overall as we see from the

results, alpha beta pruning does not affect the final output i.e. the node list to get the optimal output and we get this optimal output in lesser time complexity.