EXPERIMENT NO. 5

Aim: Write a program to implement Greedy Best First Search.

Requirements: Compatible version of python.

Theory:

The informed search algorithm is also called heuristic search or directed search. In contrast to uninformed search algorithms, informed search algorithms require details such as distance to reach the goal, steps to reach the goal, cost of the paths which makes this algorithm more efficient.

Here, the goal state can be achieved by using the heuristic function.

The heuristic function is used to achieve the goal state with the lowest cost possible. This function estimates how close a state is to the goal.

Algorithm:

Greedy best-first search uses the properties of both depth-first search and breadth-first search. Greedy best-first search traverses the node by selecting the path which appears best at the moment. The closest path is selected by using the heuristic function.

Implementation:

```
from queue import PriorityQueue

def create_path(parent,dest):
    temp = []

    while(dest):
        temp.append(dest)
        dest = parent[dest]

    return list(reversed(temp))

def gbfs(graph,source,dest,heu):
```

```
    parent, visited = {}, set()
    que = PriorityQueue()
    que.put((heu[source],source))
    parent[source] = None
    while(not que.empty()):

        current = que.get()[1]

        if current == dest:
            return create_path(parent,dest)

        visited.add(current)

        for neighbour in graph[current]:

            if neighbour in visited:
                continue

            que.put((heu[neighbour],neighbour))
            parent[neighbour] = current

    return "path doen't exist"

graph = {
    "Arad":["Sibiu", "Timisoara","Zerind"],
    "Sibiu": ["Arad","Fagaras","Oradea", "RimnicuVilcea"],
    "Timisoara": ["RimnicuVilcea"],
    "Zerind": [],
    "Fagaras": ["Sibiu","Bucharest"],
    "Oradea":[],
    "RimnicuVilcea": ["Arad"],
    "Bucharest": ["Zerind"],
}

heu = {
    "Arad": 366, "Bucharest":0, "Fagaras":176,
    "Sibiu":359, "Timisoara": 253, "Zerind":350,
    "Oradea":170, "RimnicuVilcea": 100,
    "Bucharest":0
}
inputs = [
    (graph,"Arad","Bucharest",heu),
    (graph,"Arad","RimnicuVilcea",heu),
    (graph,"Arad","Sinaia",heu)
]

for x in inputs:
    print(f"path from {x[1]} to {x[2]}: {gbfs(*x)}\n")
```

Output:

```
path from Arad to Bucharest: ['Arad', 'Sibiu', 'Fagaras', 'Bucharest']

path from Arad to RimnicuVilcea: ['Arad', 'Timisoara', 'RimnicuVilcea']

path from Arad to Sinaia: path doen't exist
```

Conclusion: We have successfully implemented Greedy Best First Search in python.