Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

Course No: CSE4108

Course Title: Artificial Intelligence Lab

Assignment No: 03

Submitted by

Name: Rizeya Rabbi Reyad

ID: 160104082

Lab Group: B2

Question 1:

Write a Python program that reads the file created as demonstrated into a dictionary taking ‘name’ as the key and a list consisting of ‘dept’ and ‘cgpa’ as the value for each line. Make changes in some ‘cgpa’ and then write back the whole file.

Code

```
def readFile():
  file = open("database.txt", "r")
for line in file:
     name, dept, cgpa = line.split("\t")
myDict[name] = [dept, float(cgpa)]
file.close    print(myDict)



def writeFile():
  file = open("database.txt", "w")

  for (key, values) in myDict.items():       line = str(key) + "\t"
+ str(values[0]) + "\t" + str(values[1])       print(line, end="\n",
file=file)
   file.close
print(myDict)



# Start from here
myDict = {} readFile()

print("\nUpdate   CGPA")   name   =
str(input("Enter   Name:   "))  cgpa   =
str(input("Enter New CGPA: "))

for (key, values) in myDict.items():
if key == name:
     myDict[name] = [values[0], cgpa]

writeFile()
```

Question 2:

Implement in generic ways (as multi-modular and interactive systems) the Greedy Best-First and A* search algorithms in Prolog and in Python.

Greedy Best-First

Code 1

```
Filename: offline3_2_gbfs from heapq
import heappush, heappop

INF = int(1e18) nodeBank
= 150

par = [-1] * nodeBank
dist = [INF] * nodeBank

h = [] for i in range(0, nodeBank):    h.append(0) h[ord('i')],
h[ord('b')], h[ord('e')], h[ord('a')] = 80, 42, 20, 55
h[ord('c')], h[ord('d')], h[ord('f')], h[ord('g')] = 34, 25, 17, 0

adj = [[] for i in range(nodeBank)] adj[ord('i')].append((ord('a'), 35)),
adj[ord('a')].append((ord('i'), 35)) adj[ord('i')].append((ord('b'), 45)),
adj[ord('b')].append((ord('i'), 45)) adj[ord('c')].append((ord('a'), 22)),
adj[ord('a')].append((ord('c'), 22)) adj[ord('c')].append((ord('g'), 47)),
adj[ord('g')].append((ord('c'), 47)) adj[ord('g')].append((ord('e'), 26)),
adj[ord('e')].append((ord('g'), 26)) adj[ord('e')].append((ord('b'), 36)),
adj[ord('b')].append((ord('e'), 36)) adj[ord('f')].append((ord('b'), 27)),
adj[ord('b')].append((ord('f'), 27)) adj[ord('d')].append((ord('g'), 30)),
adj[ord('g')].append((ord('d'), 30)) adj[ord('d')].append((ord('a'), 32)),
adj[ord('a')].append((ord('d'), 32)) adj[ord('d')].append((ord('b'), 28)),
adj[ord('b')].append((ord('d'), 28)) adj[ord('d')].append((ord('c'), 31)),
adj[ord('c')].append((ord('d'), 31))



def printPath(goal):
  path = []    while
goal != -1:
      path.append(goal)
      goal = par[goal]
path.reverse()    for v
in path:
      print(chr(v), end=" ")
```

```
def gBFS(start, goal):    pq = [(0, 0,
start)]    dist[start] = 0    visited =
[start]    while len(pq) > 0:        hf, d, u
= heappop(pq)        if goal == u:
print("Path: ", end=' ')
printPath(goal)            print()
return dist[u]        for edge in adj[u]:
w, v = edge[1], edge[0]            if v not
in visited:                dist[v] = dist[u] +
w                par[v] = u
visited.append(v)
heappush(pq, (h[v], dist[v], v))    return
INF


def run():
   start = input('Enter Start Node: ')
cost    =    gBFS(ord(start),    ord('g'))
print("Distance: ", cost, end='\n')
```

Code 2 (Run from here)

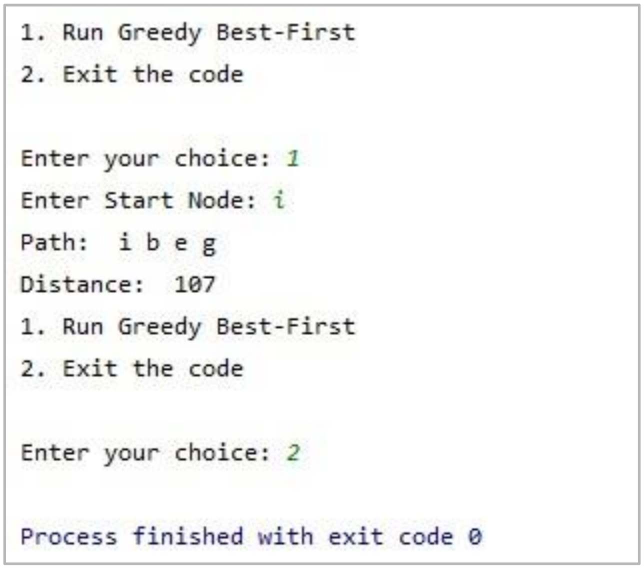
Filename: offline3_2_gbfs

```
import offline3_2_gbfs;


def display():
   print("1. Run Greedy Best-First")
print("2. Exit the code")    print()


def options():
while True:
display()

       var = input("Enter your choice: ")
if var == '1':
           offline3_2_gbfs.run()
elif var == '2':
           break
else:
           print("Wrong entry, try again")
```

```
        print()
```

```
# Start from here options()
```

Output



```
1. Run Greedy Best-First
2. Exit the code

Enter your choice: 1
Enter Start Node: i
Path:  i b e g
Distance:  107
1. Run Greedy Best-First
2. Exit the code

Enter your choice: 2

Process finished with exit code 0
```

A* search

Code 1

Filename: offline3_3_a_star

```
from
heapq import heappush, heappop

INF = int(1e18) nodeBank
= 150

dist = [INF] * nodeBank
par = [-1] * nodeBank

h = [] for i in range(0, nodeBank):    h.append(0) h[ord('i')],
h[ord('b')], h[ord('e')], h[ord('a')] = 80, 42, 20, 55
h[ord('c')], h[ord('d')], h[ord('f')], h[ord('g')] = 34, 25, 17, 0
```

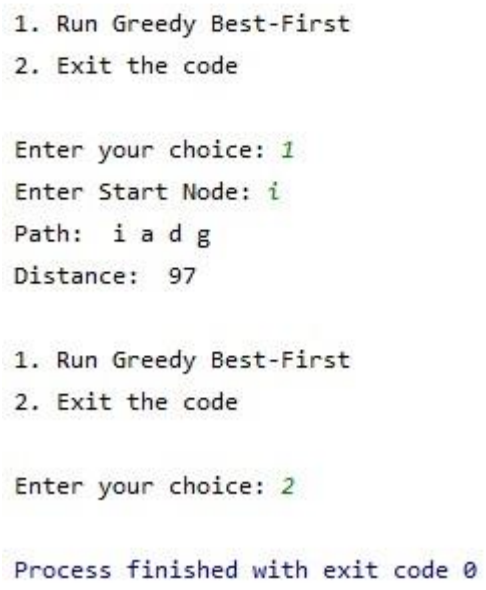
```
adj = [[] for i in range(nodeBank)] adj[ord('i')].append((ord('a'), 35)),
adj[ord('a')].append((ord('i'), 35)) adj[ord('i')].append((ord('b'), 45)),
adj[ord('b')].append((ord('i'), 45)) adj[ord('c')].append((ord('a'), 22)),
adj[ord('a')].append((ord('c'), 22)) adj[ord('c')].append((ord('g'), 47)),
adj[ord('g')].append((ord('c'), 47)) adj[ord('g')].append((ord('e'), 26)),
adj[ord('e')].append((ord('g'), 26)) adj[ord('e')].append((ord('b'), 36)),
adj[ord('b')].append((ord('e'), 36)) adj[ord('f')].append((ord('b'), 27)),
adj[ord('b')].append((ord('f'), 27)) adj[ord('d')].append((ord('g'), 30)),
adj[ord('g')].append((ord('d'), 30)) adj[ord('d')].append((ord('a'), 32)),
adj[ord('a')].append((ord('d'), 32)) adj[ord('d')].append((ord('b'), 28)),
adj[ord('b')].append((ord('d'), 28)) adj[ord('d')].append((ord('c'), 31)),
adj[ord('c')].append((ord('d'), 31))


def printPath(goal):
   path = []    while goal
!= -1:
path.append(goal)
goal = par[goal]
path.reverse()    for v in
path:
       print(chr(v), end=" ")


def aStarSearch(start, goal):
   pq = [(0, start)]
dist[start] = 0    while
len(pq) > 0:        d, u =
heappop(pq)        if u ==
goal:           print("Path:
", end=' ')
printPath(goal)
print()            return
dist[u]        for edge in
adj[u]:
           v, w = edge[0], dist[u] + edge[1]
if w < dist[v]:
               dist[v],     par[v]     =     w,     u
heappush(pq, (dist[v] + h[v], v))     return
INF


def run():
  start = input('Enter Start Node: ')
cost = aStarSearch(ord(start), ord('g'))
print("Distance: ", cost)
```

Code 2 (Run from here)

Filename: offline3_3_exe

```
import offline3_3_a_star


def display():
print()
  print("1. Run Greedy Best-First")
print("2. Exit the code")    print()


def options():
while True:
display()

     var = input('Enter your choice: ')
if var == '1':
        offline3_3_a_star.run()
elif var == '2':           break
     else:
        print("Wrong entry, try again")
        print()


options()
```

Output



```
1. Run Greedy Best-First
2. Exit the code

Enter your choice: 1
Enter Start Node: i
Path:  i a d g
Distance:  97

1. Run Greedy Best-First
2. Exit the code

Enter your choice: 2

Process finished with exit code 0
```