Jawaban UTS Praktikum Kecerdasan Buatan

Soal 1: DFS dan BFS

Representasi graf menggunakan fungsi addEdge:

Contoh kode:

```python
from collections import defaultdict

class Graph:
    def __init__(self):
        self.graph = defaultdict(list)

    def addEdge(self, u, v):
        self.graph[u].append(v)
        self.graph[v].append(u)  # Jika graf tidak berarah

    def DFS(self, start):
        visited = set()
        self._dfs_util(start, visited)

    def _dfs_util(self, v, visited):
        visited.add(v)
        print(v, end=' ')
        for neighbour in self.graph[v]:
            if neighbour not in visited:
                self._dfs_util(neighbour, visited)

    def BFS(self, start):
        visited = set()
```

Jawaban UTS Praktikum Kecerdasan Buatan

```
        queue = [start]
        visited.add(start)
        while queue:
            vertex = queue.pop(0)
            print(vertex, end=' ')
            for neighbour in self.graph[vertex]:
                if neighbour not in visited:
                    visited.add(neighbour)
                    queue.append(neighbour)

g = Graph()
g.addEdge('A', 'B')
g.addEdge('A', 'C')
g.addEdge('B', 'D')
g.addEdge('C', 'E')

print("DFS traversal:")
g.DFS('A')

print("\nBFS traversal:")
g.BFS('A')
```

Hasil running di Google Colab:

DFS traversal:
A B D C E

BFS traversal:
A B C D E

Jawaban UTS Praktikum Kecerdasan Buatan

Soal 2: Greedy dan A* Search

Contoh kode:

```python
import heapq

graph = {
    'A': [('B', 1), ('C', 4)],
    'B': [('D', 2), ('E', 5)],
    'C': [('F', 3)],
    'D': [('G', 6)],
    'E': [('G', 2)],
    'F': [('G', 1)],
    'G': []
}

heuristics = {
    'A': 7, 'B': 6, 'C': 5, 'D': 4, 'E': 3, 'F': 2, 'G': 0
}

def greedy_best_first(start, goal):
    queue = [(heuristics[start], start)]
    visited = set()
    while queue:
        h, node = heapq.heappop(queue)
        print(node, end=' ')
        if node == goal:
            print("\nGoal reached!")
            return
```

Jawaban UTS Praktikum Kecerdasan Buatan

```
        visited.add(node)
        for neighbour, cost in graph[node]:
            if neighbour not in visited:
                heapq.heappush(queue, (heuristics[neighbour], neighbour))

def a_star(start, goal):
    queue = [(heuristics[start], 0, start)]
    visited = set()
    while queue:
        est_total, cost_so_far, node = heapq.heappop(queue)
        print(node, end=' ')
        if node == goal:
            print(f"\nGoal reached! Total cost: {cost_so_far}")
            return
        visited.add(node)
        for neighbour, cost in graph[node]:
            if neighbour not in visited:
                        heapq.heappush(queue, (cost_so_far + cost + heuristics[neighbour], cost_so_far + cost,
neighbour))

print("Greedy Best-First Search:")
greedy_best_first('A', 'G')

print("\nA* Search:")
a_star('A', 'G')
```

Hasil running di Google Colab:

Greedy Best-First Search:
A B D G

Jawaban UTS Praktikum Kecerdasan Buatan

Goal reached!

A* Search:

A B D G

Goal reached! Total cost: 9

Soal 3: First Order Logic - Hadiah Liburan Dari Ayah

Jawaban:

a. Anak-anak Ayah: Lina, Miko, Nita, Oki, Pina

b. Mendapatkan buku: Lina, Nita

c. Mendapatkan mainan: Miko, Oki, Pina

d. Pina mendapatkan mainan, bukan buku.

e. Riko tidak mendapatkan apa-apa dari Ayah.

Soal 4: Sistem Pakar

Contoh kode:

```python
from experta import *

class DiagnosisPenyakit(KnowledgeEngine):
    @Rule(Fact(demam=True), Fact(lelah=True), Fact(leher_bengkak=True), Fact(mulut_kering=True))
```

Jawaban UTS Praktikum Kecerdasan Buatan

```
    def gondongan(self):
        print("Diagnosis: Gondongan")

    @Rule(Fact(sakit_perut=True), Fact(diare=True), Fact(kram=True), Fact(mual=True))
    def ibs(self):
        print("Diagnosis: Irritable Bowel Syndrome (IBS)")

    @Rule(Fact(mual=True), Fact(kembung=True), Fact(sendawa=True), Fact(sakit_tenggorokan=True))
    def gerd(self):
        print("Diagnosis: Gastroesophageal Reflux Disease (GERD)")

engine = DiagnosisPenyakit()
engine.reset()
engine.declare(Fact(mual=True), Fact(kembung=True), Fact(sendawa=True), Fact(sakit_tenggorokan=True))
engine.run()
```

Hasil running di Google Colab:

```
Diagnosis: Gastroesophageal Reflux Disease (GERD)
```