TUGAS PRAKTIKUM 8
PENJELASAN DARI ALGORITMA GRAPH “M COLORING PROBLEM”

Nama : Tegar Adimas Nugroho
NIM : G.211.22.0088
Kelompok : B1 Pagi

Coding yang digunakan :

```
import networkx as nx
import matplotlib.pyplot as plt

def m_coloring(graph, m):
    colors = {}  # Simpan pewarnaan setiap simpul di sini

    # Fungsi untuk memeriksa apakah pewarnaan simpul i aman dengan warna c
    def is_safe(node, c):
        for neighbor in graph[node]:
            if neighbor in colors and colors[neighbor] == c:
                return False
        return True

    # Fungsi untuk memilih warna untuk simpul
    def choose_color(node):
        for c in range(1, m + 1):
            if is_safe(node, c):
                return c

    # Heuristik: Pilih simpul dengan derajat tertinggi terlebih dahulu
    sorted_nodes = sorted(graph.degree, key=lambda x: x[1], reverse=True)

    # Mewarnai simpul satu per satu
    for node, _ in sorted_nodes:
        colors[node] = choose_color(node)

    return colors

def visualize(graph, colors):
    pos = nx.spring_layout(graph)
```

```
    nx.draw(graph, pos, with_labels=True, font_weight='bold', node_color=list(colors.values()),
cmap=plt.cm.rainbow)
    plt.show()

# Buat graf dengan 10 simpul dan sambungan yang lebih kompleks
G = nx.Graph()
nodes = list(range(10))
edges = [(0, 1), (0, 2), (1, 2), (1, 3), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9), (9, 0), (0, 5)]
G.add_nodes_from(nodes)
G.add_edges_from(edges)

# Jumlah warna yang digunakan
m_value = 4

# Pewarnaan simpul
coloring_result = m_coloring(G, m_value)
print("Pewarnaan simpul:", coloring_result)

# Visualisasi graf dengan pewarnaan
visualize(G, coloring_result)
```

Penjelasan Algoritmanya :

1. Inisialisasi Graf dan Parameter:
    - Graf “G” dibuat menggunakan library NetworkX dengan 10 simpul dan sambungan yang lebih kompleks.
    - Parameter “m_value” menentukan jumlah warna yang akan digunakan untuk mewarnai graf.

    Pada codingan :

    ```
    G = nx.Graph()
    nodes = list(range(10))
    edges = [(0, 1), (0, 2), (1, 2), (1, 3), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9), (9, 0), (0, 5)]
    G.add_nodes_from(nodes)
    G.add_edges_from(edges)
    m_value = 4
    ```

2. Algoritma Pewarnaan:
    - Fungsi “m_coloring” mengimplementasikan algoritma pewarnaan graf.
    - Variabel “colors” menyimpan pewarnaan setiap simpul.

- Fungsi “is_safe” memeriksa apakah pewarnaan simpul node aman dengan warna “c”.
- Fungsi “choose_color” memilih warna yang aman untuk simpul “node”.
- Simpul diurutkan berdasarkan derajatnya (jumlah tetangga) secara menurun.
- Mewarnai simpul satu per satu, dan pada setiap langkah memilih warna yang aman untuk simpul tersebut.

Pada Codingan :

```
def m_coloring(graph, m):
    colors = {}

    def is_safe(node, c):
        for neighbor in graph[node]:
            if neighbor in colors and colors[neighbor] == c:
                return False
        return True

    def choose_color(node):
        for c in range(1, m + 1):
            if is_safe(node, c):
                return c

    sorted_nodes = sorted(graph.degree, key=lambda x: x[1], reverse=True)

    for node, _ in sorted_nodes:
        colors[node] = choose_color(node)

    return colors
```

3. Visualisasi Graf
    - Fungsi visualize menggunakan NetworkX dan Matplotlib untuk menampilkan visualisasi graf dengan pewarnaan.
    - Simpul diatur menggunakan spring_layout untuk tata letak yang baik.
    - Setiap simpul diwarnai sesuai dengan hasil pewarnaan.

Pada Codingan :

```
def visualize(graph, colors):
    pos = nx.spring_layout(graph)
    nx.draw(graph, pos, with_labels=True, font_weight='bold',
node_color=list(colors.values()), cmap=plt.cm.rainbow)
    plt.show()
```

4. Penggunaan dan Output
    - Membuat pewarnaan graf dengan memanggil fungsi “m_coloring “dan menampilkan visualisasi dengan memanggil fungsi “visualize”

    Pada Codingan :

```
coloring_result = m_coloring(G, m_value)
print("Pewarnaan simpul:", coloring_result)

visualize(G, coloring_result).
```