Tugas Josheet 7

```
#include <stdio.h>
#include <stdlib.h>

#define MAX_SIZE 100

// Struktur Queue
typedef struct {
    int items[MAX_SIZE];
    int front;
    int rear;
} Antrian;

// Fungsi untuk membuat antrian baru
Antrian* buatAntrian() {
    Antrian* antrian = (Antrian*)malloc(sizeof(Antrian));
    antrian->front = -1;
    antrian->rear = -1;
    return antrian;
}

// Fungsi untuk memeriksa apakah antrian kosong
int kosong(Antrian* antrian) {
    if (antrian->rear == -1)
        return 1;
    else
        return 0;
}
```

```
// Fungsi untuk menambahkan elemen ke dalam antrian
void masukkan(Antrian* antrian, int nilai) {
    if (antrian->rear == MAX_SIZE - 1)
        printf("Antrian penuh\n");
    else {
        if (antrian->front == -1)
            antrian->front = 0;
        antrian->rear++;
        antrian->items[antrian->rear] = nilai;
    }
}

// Fungsi untuk mengeluarkan elemen dari antrian
int keluarkan(Antrian* antrian) {
    int elemen;
    if (kosong(antrian)) {
        printf("Antrian kosong\n");
        return -1;
    } else {
        elemen = antrian->items[antrian->front];
        antrian->front++;
        if (antrian->front > antrian->rear) {
            antrian->front = antrian->rear = -1;
        }
        return elemen;
    }
}

// Fungsi untuk membuat graf
int** buatGraf(int vertices) {
```

```
    int** graf = (int**)malloc(vertices * sizeof(int*));
    for (int i = 0; i < vertices; i++) {
        graf[i] = (int*)malloc(vertices * sizeof(int));
        for (int j = 0; j < vertices; j++) {
            graf[i][j] = 0;
        }
    }
    return graf;
}

// Fungsi untuk menambahkan tepian ke dalam graf
void tambahTepian(int** graf, int src, int dest) {
    graf[src][dest] = 1;
    graf[dest][src] = 1;
}

// Fungsi untuk melakukan Breadth First Search
void BFS(int** graf, int vertices, int mulai) {
    int dikunjungi[MAX_SIZE] = {0};
    Antrian* antrian = buatAntrian();

    dikunjungi[mulai] = 1;
    masukkan(antrian, mulai);

    printf("Breadth First Search dimulai dari simpul %d: ", mulai);

    while (!kosong(antrian)) {
        int simpulSaatIni = keluarkan(antrian);
        printf("%d ", simpulSaatIni);
```

```
        for (int i = 0; i < vertices; i++) {
            if (graf[simpulSaatIni][i] == 1 && !dikunjungi[i]) {
                dikunjungi[i] = 1;
                masukkan(antrian, i);
            }
        }
    }

    printf("\n");
}

int main() {
    int vertices = 6;
    int** graf = buatGraf(vertices);

    tambahTepian(graf, 0, 1);
    tambahTepian(graf, 0, 2);
    tambahTepian(graf, 1, 3);
    tambahTepian(graf, 1, 4);
    tambahTepian(graf, 2, 4);
    tambahTepian(graf, 3, 5);
    tambahTepian(graf, 4, 5);

    BFS(graf, vertices, 0);

    return 0;
}
```

Breadth First Search (BFS) adalah salah satu algoritma pencarian yang digunakan untuk melakukan penelusuran atau traversal pada graf atau pohon. Algoritma ini dimulai dari simpul awal (start node) dan menelusuri semua simpul yang terhubung dengan simpul tersebut secara berurutan.

Prinsip utama dari BFS adalah penggunaan queue. Queue digunakan untuk menyimpan simpul-simpul yang akan dieksplorasi selanjutnya. Pada setiap iterasi, simpul yang telah dieksplorasi akan dikeluarkan dari queue, kemudian simpul-simpul tetangganya akan dimasukkan ke dalam queue untuk dieksplorasi selanjutnya.

Berikut adalah bagaimana queue digunakan dalam algoritma BFS:

- Saat memulai, simpul awal dimasukkan ke dalam queue.
- Selama proses pencarian berlangsung:
- Simpul yang dikunjungi dikeluarkan dari depan queue, sehingga simpul yang pertama dimasukkan akan menjadi yang pertama dikeluarkan (FIFO - First-In-First-Out).
- Setelah simpul dikeluarkan, semua simpul terhubung yang belum dikunjungi dimasukkan ke dalam queue.
- Dengan cara ini, BFS memastikan bahwa simpul-simpul yang terhubung langsung dengan simpul awal akan dikunjungi terlebih dahulu sebelum melanjutkan ke simpul-simpul yang lebih jauh.
- Queue memastikan bahwa urutan kunjungan simpul-simpul diatur dengan baik, sehingga BFS dapat menemukan solusi yang optimal dalam kasus di mana solusi tersebut ada.