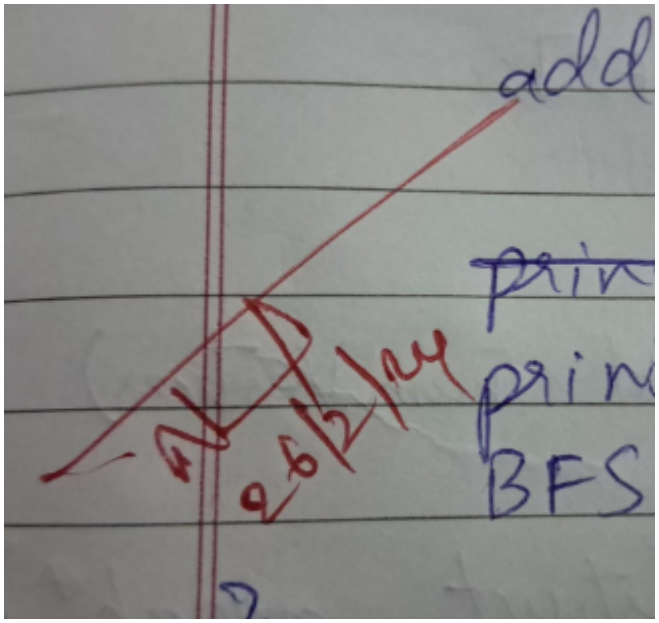
```c
#include <stdio.h>
#include <stdlib.h>

#define MAX_VERTICES 100

struct Node {
    int vertex;
    struct Node* next;
};

struct Graph {
    struct Node* adjLists[MAX_VERTICES];
    int visited[MAX_VERTICES];
};

struct Node* createNode(int v) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->vertex = v;
    newNode->next = NULL;
    return newNode;
}

struct Graph* createGraph() {
    struct Graph* graph = (struct Graph*)malloc(sizeof(struct Graph));
    int i;
    for (i = 0; i < MAX_VERTICES; i++) {
        graph->adjLists[i] = NULL;
        graph->visited[i] = 0;
    }
    return graph;
}

void addEdge(struct Graph* graph, int src, int dest) {
    struct Node* newNode = createNode(dest);
    newNode->next = graph->adjLists[src];
    graph->adjLists[src] = newNode;

    newNode = createNode(src);
    newNode->next = graph->adjLists[dest];
    graph->adjLists[dest] = newNode;
}

void BFS(struct Graph* graph, int startVertex) {
```

```
    struct Node* temp;
    int queue[MAX_VERTICES];
    int front = 0, rear = 0, v;

    graph->visited[startVertex] = 1;
    queue[rear++] = startVertex;

    while (front < rear) {
        v = queue[front++];
        printf("%d ", v);

        for (temp = graph->adjLists[v]; temp != NULL; temp =
temp->next) {
            if (!graph->visited[temp->vertex]) {
                graph->visited[temp->vertex] = 1;
                queue[rear++] = temp->vertex;
            }
        }
    }
}

void main() {
    struct Graph* graph = createGraph();

    addEdge(graph, 0, 1);
    addEdge(graph, 0, 2);
    addEdge(graph, 1, 2);
    addEdge(graph, 2, 3);
    addEdge(graph, 1, 3);

    printf("Breadth First Traversal starting from vertex 0: ");
    BFS(graph, 0);
}
```

```
Breadth First Traversal starting from vertex 0: 0 2 1 3
```


add
~~prin~~
prin
BFS