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

#define MAX_NODES 100

struct Edge {
    int source;
    int destination;
    int weight;
};

struct PQNode {
    int node;
    int distance;
};

int compare(const void* a, const void* b) {
    const struct PQNode* nodeA = (const struct PQNode*)a;
    const struct PQNode* nodeB = (const struct PQNode*)b;
    return nodeA->distance - nodeB->distance;
}

int dijkstra(int n, int k, struct Edge* edges, int num_edges) {
    // Initialize the distance array with infinity (INT_MAX)
    int distances[MAX_NODES];
    for (int i = 1; i <= n; i++) {
        distances[i] = INT_MAX;
    }
    distances[k] = 0;

    struct PQNode pq[MAX_NODES];
    int pq_size = 0;
    pq[pq_size++] = (struct PQNode){k, 0};

    while (pq_size > 0) {
        // Get the node with the shortest distance from the source
        int node = pq[0].node;
        int distance = pq[0].distance;
        pq[0] = pq[--pq_size];
        for (int i = 0; i < num_edges; i++) {
            struct Edge edge = edges[i];
            if (edge.source == node) {
                int neighbor = edge.destination;
                int neighbor_distance = distance + edge.weight;
                if (neighbor_distance < distances[neighbor]) {
                    distances[neighbor] = neighbor_distance;
                    pq[pq_size++] = (struct PQNode){neighbor, neighbor_distance};
                    qsort(pq, pq_size, sizeof(struct PQNode), compare);
                }
            }
        }
    }

    int max_distance = 0;
    for (int i = 1; i <= n; i++) {
        if (distances[i] == INT_MAX) {
            return -1;
        }
        if (distances[i] > max_distance) {
            max_distance = distances[i];
        }
    }

    return max_distance;
}

int main() {
    int n, num_edges, k;
    scanf("%d %d %d", &n, &num_edges, &k);

    struct Edge edges[num_edges];
    for (int i = 0; i < num_edges; i++) {
        int u, v, w;
        scanf("%d %d %d", &u, &v, &w);
        edges[i] = (struct Edge){u, v, w};
    }

    int result = dijkstra(n, k, edges, num_edges);
    printf("%d\n", result);

    return 0;
}
```