```c
/*
Floyd - Warshall tim duong di ngan nhat giua cac cap dinh
in chieu dai giua cac cap dinh.
*/
#include <stdio.h>

#define MAXN 1000
#define NO_EDGE 0
#define INFINITY 9999999

// Graph
typedef struct {
    int n, m;
    int edges[MAXN][MAXN];
} Graph;

void init_graph(Graph* G, int n) {
    G->n = n;
    G->m = 0;

    int i, j;
    for (i = 1; i <= n; ++i) {
        for (j = 1; j <= n; ++j) {
            G->edges[i][j] = NO_EDGE;
        }
    }
}

void add_edge(Graph* G, int u, int v, int w) {
    G->edges[u][v] = w;
    ++G->m;
}

int pi[MAXN][MAXN];
int next[MAXN][MAXN];

void FloydWarshall(Graph* G) {
    int u, v, k;
    for (u = 1; u <= G->n; ++u) {
        for (v = 1; v <= G->n; ++v) {
            pi[u][v] = INFINITY;
            next[u][v] = -1;
        }
```

```
    }

    for (u = 1; u <= G-> n; ++u) {
        pi[u][u] = 0;
    }

    for (u = 1; u <= G->n; ++u) {
        for (v = 1; v <= G->n; ++v) {
            if (G->edges[u][v] != NO_EDGE) {
                pi[u][v] = G->edges[u][v];
                next[u][v] = v;
            }
        }
    }

    for (k = 1; k <= G->n; ++k) {
        for (u = 1; u <= G->n; ++u) {
            for (v = 1; v <= G->n; ++v) {
                if ((pi[u][k] + pi[k][v] < pi[u][v]) &&
                    (pi[u][k] != INFINITY) && (pi[k][v] != INFINITY))
{
                    pi[u][v] = pi[u][k] + pi[k][v];
                    next[u][v] = next[u][k];
                }
            }
        }
    }
}

int main() {
    Graph G;
    int n, m, u, v, w, e, s;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

    for (e = 0; e < m; e++) {
        scanf("%d%d%d", &u, &v, &w);
        add_edge(&G, u, v, w);
    }

    FloydWarshall(&G);

    int i, j;
```

```
    for (i = 1; i <= n; ++i) {
        for (j = 1; j <= n; ++j) {
            if (pi[i][j] == INFINITY) {
                printf("%d -> %d: oo\n", i, j);
            } else {
                printf("%d -> %d: %d\n", i, j, pi[i][j]);
            }
        }
    }
    return 0;
}
```