```c
// Kiem tra chu trinh am va dung dung duong di ngan nhat 1
#include <stdio.h>

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

// Graph
typedef struct {
    int u, v;
    int w;
} Edge;
typedef struct {
    int n, m;
    Edge edges[1000];
} Graph;

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

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

int pi[MAXN];
int p[MAXN];
int negative_cycle = 0;

void BellmanFord(Graph* G, int s) {
    int i, j, it;
    for (i = 1; i <= G->n; ++i) {
        pi[i] = INFINITY;
    }

    pi[s] = 0;
    p[s] = -1;

    for (it = 1; it < G->n; ++it) {
```

```
        for (j = 0; j < G->m; ++j) {
            int u = G->edges[j].u;
            int v = G->edges[j].v;
            int w = G->edges[j].w;
            if (pi[u] + w < pi[v]) {
                pi[v] = pi[u] + w;
                p[v] = u;
            }
        }
    }

    // Kiem tra lan nua neu co the cap nhat thi chu trinh am
    for (j = 0; j < G->m; ++j) {
        int u = G->edges[j].u;
        int v = G->edges[j].v;
        int w = G->edges[j].w;
        if (pi[u] + w < pi[v]) {
            negative_cycle = 1;
        }
    }

}

int main() {
    Graph G;
    int n, m, u, v, w, e;
    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);
    }

    BellmanFord(&G, 1);

    if (negative_cycle == 1) {
        printf("negative cycle");
    } else {
        printf("ok");
    }
    return 0;
}
```