Started on	Tuesday, 24 June 2025, 10:41 PM
State	Finished
Completed on	Tuesday, 24 June 2025, 10:41 PM
Time taken	11 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

Viết chương trình đọc vào một **đơn đồ thị có hướng, có trọng số**, kiểm tra xem nó có chứa chu trình âm hay không.

Đầu vào (Input)

Dữ liệu đầu vào được nhập từ dòng nhập chuẩn (bàn phím, stdin) với định dạng:

- Dòng đầu tiên chứa 2 số nguyên n và m tương ứng là số đỉnh và số cung.
- m dòng tiếp theo mỗi dòng chứa 3 số nguyên u, v, w nói rằng cung (u, v) có trọng số w.

Đầu ra (Output)

- In ra màn hình **YES** nếu phát hiện có chu trình âm, ngược lại in ra **NO**.
- Xem thêm ví dụ bên dưới.

For example:

Input	Result
4 4 1 2 1 2 3 -1 3 4 -1 4 1 -1	YES
8 13 1 2 4 1 3 4 3 5 4 3 6 -2 4 1 3 4 3 2 5 4 1 5 7 -2 6 2 3 6 5 -3 7 6 2 7 8 2 8 5 -2 1	YES
8 13 1 2 4 1 3 4 3 5 4 3 6 2 4 1 3 4 3 2 5 4 1 5 7 5 6 2 3 6 5 -3 7 6 2 7 8 2 8 5 -2 1	NO

Answer: (penalty regime: 0 %)

```
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
```

```
}

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


#define oo 999999
int pi[MAXN][MAXN];
int next[MAXN][MAXN];
int FloydWarshall(Graph *pG) {
    int u, v, k;
    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++) {
            pi[u][v] = oo;
            next[u][v] = -1;
        }

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

    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++)
            if (pG->W[u][v] != NO_EDGE) {
                pi[u][v] = pG->W[u][v]; //đi trực tiếp từ u -> v
                next[u][v] = v;
            }

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

    //Kiểm tra chu trình âm (nếu cần thiết)
    int negative_cycle = 0;
    for (u = 1; u <= pG->n; u++) {
        if (pi[u][u] < 0) {
            //Đồ thị có chứa chu trình âm
            negative_cycle = 1;
            break;
        }
    }
    return negative_cycle;
}


int main() {
    Graph G;
    int n, m;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

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

    if (FloydWarshall(&G) == 1)
        printf("YES\n");
    else
        printf("NO\n");


    return 0;
}
```

Debug: source code from all test runs

Run 1

```
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
}

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


#define oo 999999
int pi[MAXN][MAXN];
int next[MAXN][MAXN];
int FloydWarshall(Graph *pG) {
    int u, v, k;
    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++) {
            pi[u][v] = oo;
            next[u][v] = -1;
        }

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

    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++)
            if (pG->W[u][v] != NO_EDGE) {
                pi[u][v] = pG->W[u][v]; //đi trực tiếp từ u -> v
                next[u][v] = v;
            }

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

    //Kiểm tra chu trình âm (nếu cần thiết)
    int negative_cycle = 0;
    for (u = 1; u <= pG->n; u++) {
        if (pi[u][u] < 0) {
            //Đồ thị có chứa chu trình âm
            negative_cycle = 1;
            break;
        }
    }
    return negative_cycle;
}


int main() {
    Graph G;
    int n, m;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

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

```
    if (FloydWarshall(&G) == 1)
        printf("YES\n");
    else
        printf("NO\n");


    return 0;
}
```

Run 2

```
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
}

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


#define oo 999999
int pi[MAXN][MAXN];
int next[MAXN][MAXN];
int FloydWarshall(Graph *pG) {
    int u, v, k;
    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++) {
            pi[u][v] = oo;
            next[u][v] = -1;
        }

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

    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++)
            if (pG->W[u][v] != NO_EDGE) {
                pi[u][v] = pG->W[u][v]; //đi trực tiếp từ u -> v
                next[u][v] = v;
            }

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

    //Kiểm tra chu trình âm (nếu cần thiết)
    int negative_cycle = 0;
    for (u = 1; u <= pG->n; u++) {
        if (pi[u][u] < 0) {
            //Đồ thị có chứa chu trình âm
            negative_cycle = 1;
            break;
        }
    }
    return negative_cycle;
}


int main() {
    Graph G;
    int n, m;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

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

```
    if (FloydWarshall(&G) == 1)
        printf("YES\n");
    else
        printf("NO\n");


    return 0;
}
```

Run 3

```
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
}

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


#define oo 999999
int pi[MAXN][MAXN];
int next[MAXN][MAXN];
int FloydWarshall(Graph *pG) {
    int u, v, k;
    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++) {
            pi[u][v] = oo;
            next[u][v] = -1;
        }

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

    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++)
            if (pG->W[u][v] != NO_EDGE) {
                pi[u][v] = pG->W[u][v]; //đi trực tiếp từ u -> v
                next[u][v] = v;
            }

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

    //Kiểm tra chu trình âm (nếu cần thiết)
    int negative_cycle = 0;
    for (u = 1; u <= pG->n; u++) {
        if (pi[u][u] < 0) {
            //Đồ thị có chứa chu trình âm
            negative_cycle = 1;
            break;
        }
    }
    return negative_cycle;
}


int main() {
    Graph G;
    int n, m;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

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

```
    if (FloydWarshall(&G) == 1)
        printf("YES\n");
    else
        printf("NO\n");


    return 0;
}
```

Run 4

```
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
}

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


#define oo 999999
int pi[MAXN][MAXN];
int next[MAXN][MAXN];
int FloydWarshall(Graph *pG) {
    int u, v, k;
    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++) {
            pi[u][v] = oo;
            next[u][v] = -1;
        }

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

    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++)
            if (pG->W[u][v] != NO_EDGE) {
                pi[u][v] = pG->W[u][v]; //đi trực tiếp từ u -> v
                next[u][v] = v;
            }

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

    //Kiểm tra chu trình âm (nếu cần thiết)
    int negative_cycle = 0;
    for (u = 1; u <= pG->n; u++) {
        if (pi[u][u] < 0) {
            //Đồ thị có chứa chu trình âm
            negative_cycle = 1;
            break;
        }
    }
    return negative_cycle;
}


int main() {
    Graph G;
    int n, m;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

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

```
    if (FloydWarshall(&G) == 1)
        printf("YES\n");
    else
        printf("NO\n");


    return 0;
}
```

Run 5

```
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
}

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


#define oo 999999
int pi[MAXN][MAXN];
int next[MAXN][MAXN];
int FloydWarshall(Graph *pG) {
    int u, v, k;
    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++) {
            pi[u][v] = oo;
            next[u][v] = -1;
        }

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

    for (u = 1; u <= pG->n; u++)
        for (v = 1; v <= pG->n; v++)
            if (pG->W[u][v] != NO_EDGE) {
                pi[u][v] = pG->W[u][v]; //đi trực tiếp từ u -> v
                next[u][v] = v;
            }

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

    //Kiểm tra chu trình âm (nếu cần thiết)
    int negative_cycle = 0;
    for (u = 1; u <= pG->n; u++) {
        if (pi[u][u] < 0) {
            //Đồ thị có chứa chu trình âm
            negative_cycle = 1;
            break;
        }
    }
    return negative_cycle;
}


int main() {
    Graph G;
    int n, m;
    scanf("%d%d", &n, &m);
    init_graph(&G, n);

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

```
    if (FloydWarshall(&G) == 1)
        printf("YES\n");
    else
        printf("NO\n");


    return 0;
}
```

	Input	Expected	Got	
✔	4 4 1 2 1 2 3 -1 3 4 -1 4 1 -1	YES	YES	✔
✔	8 13 1 2 4 1 3 4 3 5 4 3 6 -2 4 1 3 4 3 2 5 4 1 5 7 -2 6 2 3 6 5 -3 7 6 2 7 8 2 8 5 -2 1	YES	YES	✔
✔	8 13 1 2 4 1 3 4 3 5 4 3 6 2 4 1 3 4 3 2 5 4 1 5 7 5 6 2 3 6 5 -3 7 6 2 7 8 2 8 5 -2 1	NO	NO	✔
✔	8 14 1 2 4 1 3 4 5 3 4 6 3 -2 4 1 3 4 3 2 5 4 1 5 7 -2 6 2 3 6 5 -3 7 6 2 7 8 2 5 8 -2 4 8 -4 4	YES	YES	✔

	Input	Expected	Got	
✔	8 14 1 2 4 1 3 4 5 3 4 6 3 -2 4 1 3 4 3 2 5 4 1 5 7 -2 6 2 3 6 5 -3 7 6 2 7 8 2 5 8 -2 4 8 -4 6	YES	YES	✔

Passed all tests! ✔

Question author's solution (C):

```c
#include <stdio.h>

#define MAXM 500
#define MAXN 100
#define oo 999999
#define NO_EDGE -999999

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

void init_graph(Graph *pG, int n) {
    pG->n = n;
    pG->m = 0;
    for (int u = 1; u <= n; u++)
        for (int v = 1; v <= n; v++)
            pG->W[u][v] = NO_EDGE;
}

void add_edge(Graph *pG, int u, int v, int w) {
    pG->W[u][v] = w;
```

Correct

Marks for this submission: 1.00/1.00.

◄ Bài tập 9 - Thuật toán Floyd - Warshall (đường đi ngắn nhất giữa các cặp đỉnh)

Jump to...

Thuật toán Moore - Dijkstra ►