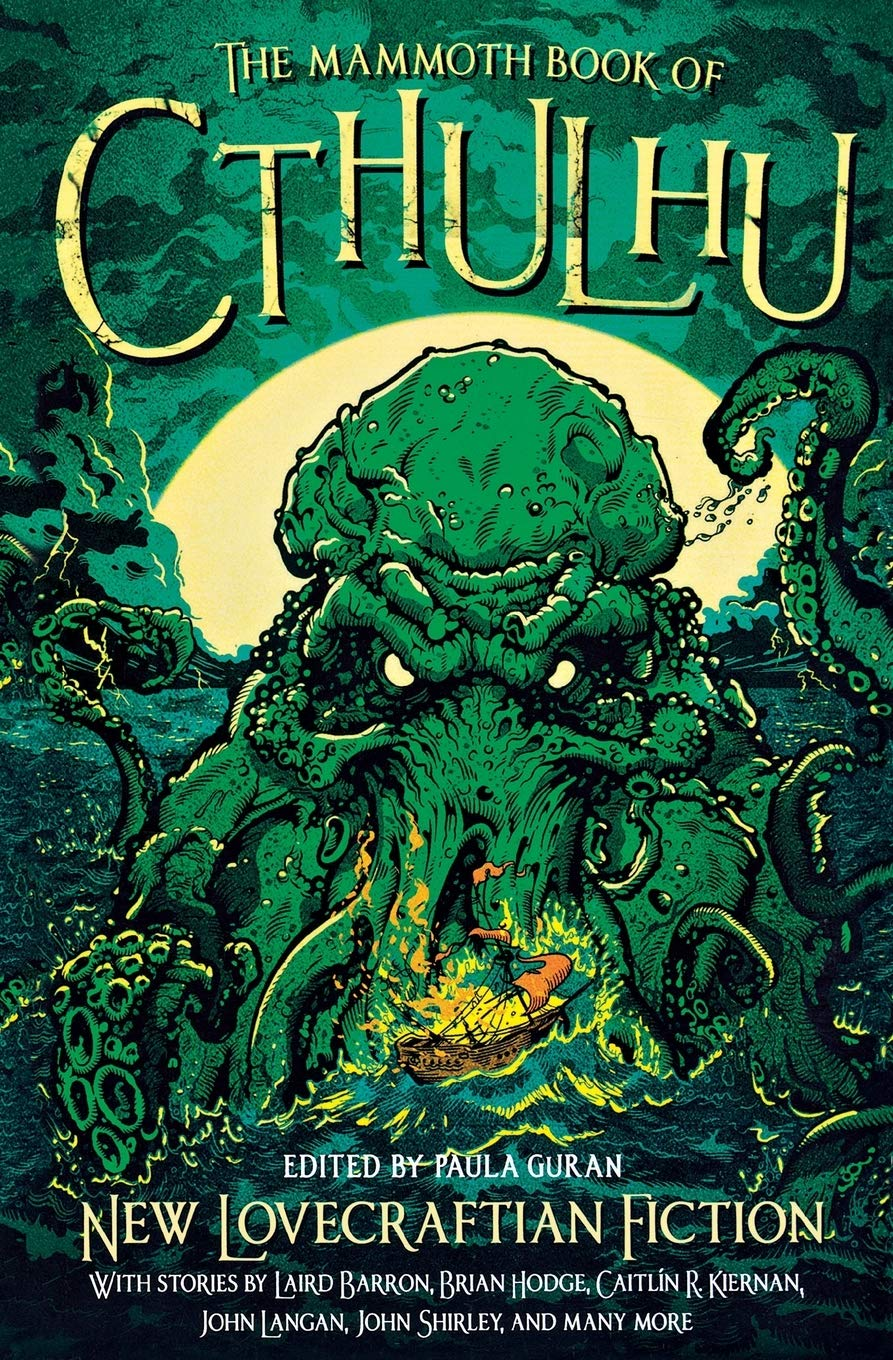
****

**Ví dụ**

**Input**

6 6

6 3

6 4

5 1

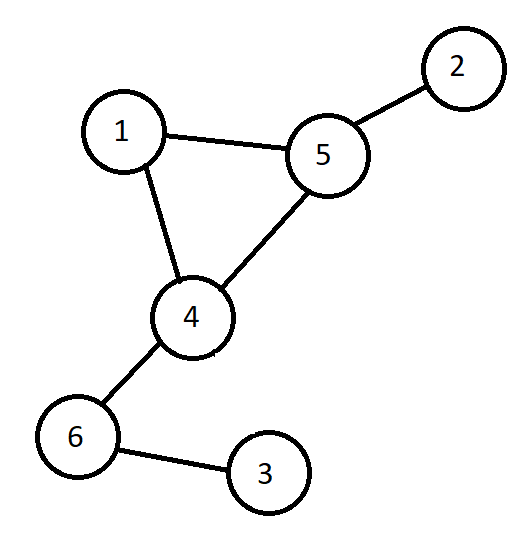
2 5

1 4

5 4

**Output**

FHTAGN!



**Input**

6 5

5 6

4 6

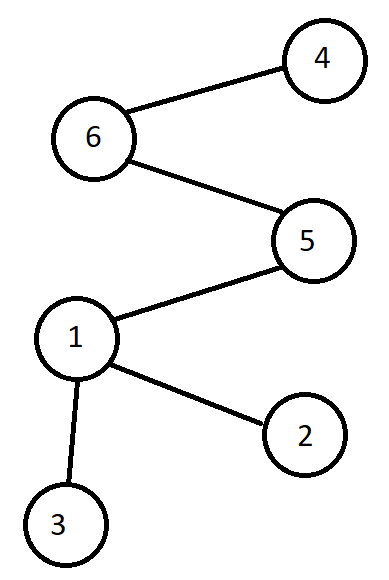
3 1

5 1

1 2

**Output**

NO



Một con Cthulhu:

* Có K cây, K >= 3
* K cây đó được kết nối bởi 1 chu trình đơn giản

Giả sử K cây, mỗi cây có số đỉnh là: k1, k2, …, kK

(k1 + k2 + … + kK = n)

Cây thứ nhất có k1 - 1 cạnh

Cây thứ 2 có k2 - 1 cạnh

…

Cây thứ kK có kK - 1 cạnh

Tổng cạnh trong K cây: k1 - 1 + k2 - 1 + … + kK - 1 = n - K

K cây được kết nối bởi 1 chu trình: → K cạnh

→ Số lượng cạnh trong đồ thị Cthulu = n

**Điều kiện là một Cthulu:**

1. Có n == m
2. Đồ thị liên thông

**Giải thuật**

**Bước 1:** Nhập dữ liệu

**Bước 2:** Kiểm tra xem số đỉnh có bằng số cạnh hay không, nếu bằng, tiếp tục bước 3, ngược lại, đến bước 4

**Bước 3:** Kiểm tra đồ thị liên thông, có thể sử dụng DSU hoặc BFS/DFS

**Bước 4:** Kết luận và kết thúc chương trình.

**Mã giả:**

**function findSet(u):**

**if parent[u] != u:**

**parent[u] = findSet(parent[u]);**

**return parent[u];**

**function unionSet(u, v):**

**up = findSet(u);**

**vp = findSet(v);**

**parent[vp] = up;**

**main():**

**read(n, m);**

**if n != m:**

**print(“NO”);**

**exit();**

**parent = []; // n + 1 Phần tử**

**for i = 1 to n:**

**parent[i] = i;**

**for i = 1 to m:**

**read(u, v);**

**unionSet(u, v);**

**countGroup = 0;**

**for i = 1 to n:**

**if parent[i] == i:**

**countGroup += 1;**

**if countGroup > 1:**

**print(‘NO’);**

**else:**

**print(‘FHTAGN!’);**

**Độ phức tạp: O(M \* log(N))**

**M là số lượng cạnh**

**N là số lượng đỉnh**