# Board Game (boardgame2)

Alice and Bob, a sibling pair, have been blessed with a weighted directed acyclic graph with $N$ nodes and $M$ edges under the Christmas tree. They are super competitive, so they decided to play a game on the graph.

Figure 1: Alice and Bob arguing as normal siblings do.

The game begins with a marker placed on vertex 1. At every turn, Alice and Bob make a move alternately with Alice beginning. In a move, the current player can move the marker from vertex $u$ to vertex $v$ if there is an edge from $u$ to $v$. For such a move, the player needs to pay $W_{u->v}$ money. The player who can not move the marker loses. They are incredibly smart, so they immediately understand who will win the game if they play optimally. Thus, the loser (being a sore loser) will do everything in their power to get the winner to pay as much money as possible. The loser does not care how much they have to pay.

Given the graph, your task is to find who will win the game and the minimum amount the winner has to pay if both siblings play optimally.

☞ Among the attachments of this task you may find a template file `boardgame2.*` with a sample incomplete implementation.

## Input

The input file consists of:

- a line containing integers $N$, $M$.

- $M$ lines, the $i$-th of which consisting of integers $U_i$, $V_i$, $W_i$, representing an edge from node $U_i$ to node $V_i$ with a weight of $W_i$.

## Output

The output file must consists of two lines:

- The first line should consist of a single string which is `Alice` or `Bob`, the winner of the game.
- The second line should consist of a single integer, the minimum amount of money the winner needs to spend.

## Constraints

- $1 \leq N \leq 100\,000$.
- $1 \leq M \leq 200\,000$.
- $1 \leq U_i, V_i \leq N$ for each $i = 0 \ldots M-1$.
- There is at most one edge between every pair of nodes, that is, each $(U_i, V_i)$ pair is unique.
- The edges form a directed acyclic graph.
- $0 \leq W_i \leq 1\,000\,000\,000$ for each $i = 0 \ldots M-1$.

## Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

| | |
|---|---|
| – **Subtask 1** (0 points) | Examples. |
| – **Subtask 2** (10 points) | $N \leq 10, M \leq 20$. |
| – **Subtask 3** (15 points) | $W_i = 0$ for each $i = 0 \ldots M-1$. |
| – **Subtask 4** (40 points) | $W_i = 1$ for each $i = 0 \ldots M-1$. |
| – **Subtask 5** (35 points) | No additional limitations. |

## Examples

| input | output |
|---|---|
| `8 8` | `Alice` |
| `1 2 1` | `7` |
| `1 5 8` | |
| `2 3 3` | |
| `1 4 2` | |
| `4 5 4` | |
| `4 6 5` | |
| `5 7 5` | |
| `6 8 4` | |

## Explanation

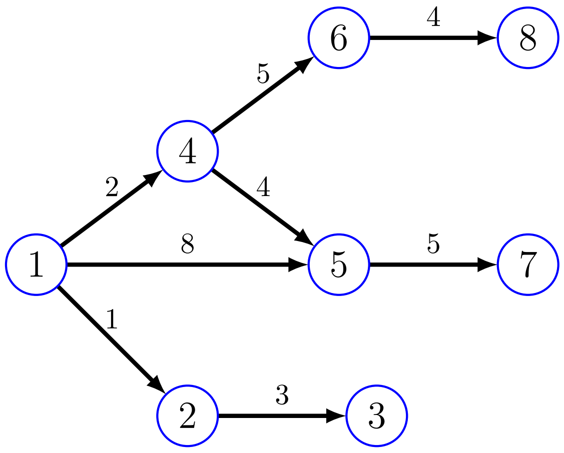

Figure 2: The graph in the sample case

In the **sample case**:

- The only way for Alice to secure her victory is to move the marker from vertex 1 to vertex 4 (for a cost of 2).
- To maximize Alice's payment, Bob will move the marker from vertex 4 to vertex 5;
- making Alice pay 5 on her last move to vertex 7.

Since Bob is unable to move, Alice wins, and she payed a total of 7.