Question:

Dynamic Programming: Good Sequence

There are **N** towers. The height of the i^{th} tower is H_i. A sequence of towers is **Good** if there are not any two adjacent towers that have the same
N) $H_{i-1} \neq H_i$ condition must hold.
You can increase the height of i^{th} tower but it will cost M_i, to increase the height by **1**.
Find the minimum cost to make the sequence good.

Note

You can increase the height of a tower any number of times you want. You have to just minimize the cost to make the sequence of towers **Good**.

Function Description

In the provided code snippet, implement the `minCost(...)` method and print the minimum cost to make the sequence good. You can write coc
phrase "`WRITE YOUR LOGIC HERE`".

There will be multiple test cases running so the Input and Output should match exactly as provided.
The base Output variable `result` is set to a default value of `-404` which can be modified. Additionally, you can add or remove these output varia

Input Format

The first line of input consists of an integer **N**.
The next **N** lines of input contain the description of towers. The i^{th} line contains H_i and M_i - the height of i^{th} tower and the cost to increase the h
respectively.

Sample Input

```
3          --denotes N.
2 4        --denotes two integers Hi and Mi
2 1        --denotes two integers Hi and Mi
3 5        --denotes two integers Hi and Mi
```

Constraints

$1 \leq N \leq 10^5$
$1 \leq H_i, M_i \leq 10^9$

Output Format

The output contains a single integer denoting the minimum cost to make a sequence Good.

Sample Output

```
2
```

Explanation

In the sample input, you have to increase the height of the second tower by 2.
Cost = $M_2 + M_2 = 1 + 1 = 2$.
Hence, the minimum cost to make the sequence Good is ***2***.

Test Case Input

```
5
2 3 7 9 10
```

Expected Output

```
2
```

Test Case Input

```
5
2 3 4 6 9
```

Expected Output

```
4
```

Test Case Input

```
4
1 2 4 6
```

Expected Output

```
3
```

Test Case Input

1
1

Expected Output

1

Test Case Input

```
10
2 3 4 5 6 8 9 10 17 92
```

Expected Output

```
6
```

Test Case Input

```
10
2 3 45 67 89 101 234 567 890 1234
```

Expected Output

```
5
```