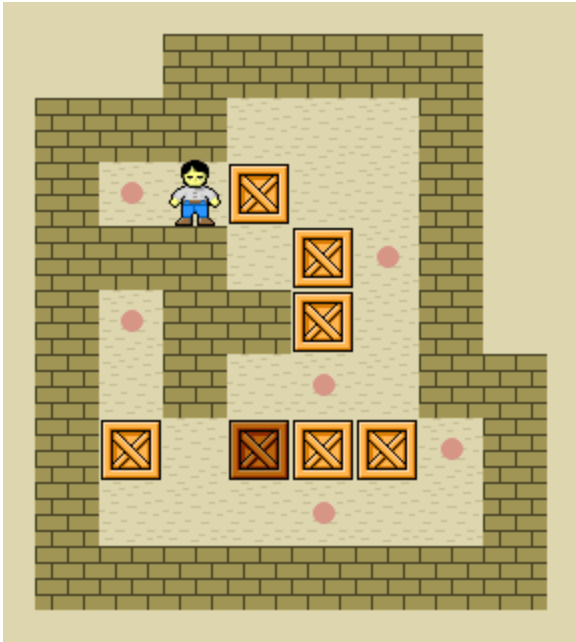


GFSSOC '15 Winter S2 - Soko-Boop

Time limit: 1.0s **Memory limit:** 128M

When Butane was still a human, one of his favourite games to play was Sokoban. In Sokoban, you play as a warehouse keeper, pushing boxes to designated spots, trying for the least number of moves. Butane attempted to program himself with a Sokoban solver so that he could fulfill his lifelong dream of becoming a warehouse keeper, but unfortunately the program is bugged!



It would be unreasonable to ask for a full Sokoban solver so your task is to code a simpler version of the game. You will be given a grid of size R by C (rows and columns), representing the Sokoban level. The grid will contain a **single box**, a **single warehouse keeper**, a **single storage location**, floor tiles and walls. The rules of our simpler game are listed below.

1. The warehouse keeper, each turn, can move one square in one of the 4 cardinal directions: up, down, left or right.
2. The warehouse keeper cannot pass through the box or walls, nor go outside of the grid. The box cannot move through walls nor go outside the grid.
3. If the warehouse keeper attempts to move through a box, the box will be pushed one square in the direction the keeper was moving, unless it will be pushed into a wall (in which case neither the keeper nor the box will move).
4. The level is solved when the box is on the storage location.
5. The goal is to solve the level with the least number of moves.

Can you help reprogram ButaneBot's simple Sokoban solver?

Input Specification

Line 1: Two integers, R and C ($1 \leq R, C \leq 30$).

The next R lines will contain the grid.

`.` characters represent the floor, `#` characters represent walls, the `B` character represents the starting location of the box, the `P` character represents the starting location of the player and `X` represents the storage location.

Output Specification

Output one integer, the minimum number of moves required to solve the level, or `-1` if the level is unsolvable.

Sample Input

```
5 5
P..#X
...#.
.B.#.
.....
..#..
```

Sample Output

```
13
```

Explanation for Sample Output

One of the optimal move sets is DOWN, RIGHT, DOWN, LEFT, DOWN, RIGHT, RIGHT, RIGHT, DOWN, RIGHT, UP, UP, UP.