Sokoban

User Guide

Matthew Falcone - CSC258

Welcome to Sokoban!

Sokoban is a puzzle game where you slide boxes onto targets on a 2D board. If you can slide every box onto a target, then you win!

Getting Started

Copy the contents of Sokoban.s to your clipboard. Then, head over to CPUlator and paste the contents into the "editor" pane. Make sure to delete the default text in the editor first.

To play the game, you will first need to assemble the program. This can be done by either clicking "Compile and Load" at the top of the editor pane, or by pressing F5.

Then, click "Continue" in the top bar to run the game, or press F3. The game will start in the terminal. The game can be restarted by clicking the "Reload" button followed by "Restart" and then "Continue".

Gameplay

Sokoban is played on a 2D board that is printed to the terminal.

Initialization - when the game is run, the user will be required to specify four parameters:

- Board width (# of rows)
- Board height (# of columns)
- # of targets (Upper bounded based on the board size, to ensure solvability)
- # of players (Must be greater than zero)

The **Inputs** section has info on how to format these parameters.

Board generation - Locations for targets and boxes are generated randomly.

Walls - Walls are used to represent the edges of the board, and are represented with the "|" character. A gap in between two "|" characters represents a row of the board.

Player - A player is represented with a "p" (lowercase). The player is placed in a random location on the board and can move up, down, left or right (see **Input)**. The player can move onto a target, in which case they become represented by a "P" (uppercase). Additionally, when the player is next to a box, moving in its direction will "push" the box in that direction. A player can move from row to row, but may not move past a wall, or otherwise try to escape the board.

Target - A target is represented with a "T". When a box is pushed onto a target, it becomes an "M", to indicate it has "matched" with a box.

Box - A box is represented with a "B". If a box is matched with a target, it can also be pushed off. The amount of boxes on the board is always the same as the number of targets.

Solving the Puzzle - To solve a board, and thus win Sokoban, every target must become matched with a box. Every time the player moves, a status message is printed which states the amount of targets which have not yet been matched.

Unsolvability - A board will always be generated in a solvable state. However, the player may end up in a situation where the game becomes unsolvable. This situation is known as a "deadlock", and it is the result of pushing a box into an area in which it cannot be moved out of. For these situations, the user may reset the board back to its original state at generation, and try again (see **Inputs** for information on how to reset).

Multiplayer - A user can play by themselves by inputting a value of 1 when asked for the number of players during initialization. Additionally, multiple players can participate by inputting a value greater than one. The game will give each player a chance to solve the same puzzle, and will indicate whose turn it is with a message when a player's turn becomes available. Players are identified with a number starting from one, and it is the responsibility of the players to designate numbers for themselves. Once all players have finished, a leaderboard will be printed, ranking players by the amount of moves they need to solve the puzzle. (Note: the printing of the leaderboard signals the end of the game.)

```
### LEADERBOARD ###
Player 1: 9 moves
Player 2: 15 moves
Player 3: 19 moves
```

Example of a leaderboard with three players, sorted by least amount of moves

Inputs

This section documents how input should be provided by the user.

Inputs can be directed into the terminal by focusing on it, i.e. clicking on the terminal pane. The

four initialization parameters should be inputted as whole numbers, and should also be within the bounds printed by the game. A bound will be specified through the terminal during initialization. If a bound is specified, the game will reject inputs outside of this bound until it receives a valid input.

Player Movement - Once the first board has been printed, Player can be moved using WASD on the keyboard - W moves up, A moves down, S moves left, and D moves right.

Resetting the board - The user can press R at any time after the game has started if they wish to reset the position of Player, box and targets back to their initial state that was created during board generation. If the player uses this feature, their number of moves will remain unchanged.

```
Welcome to Sokoban!
Enter board width (Min 3): 5
Enter board height (Min 3): 5
Enter # of targets. Max: 2
2
Generating board...
Enter # of players. 1
Your turn, Player 1

| T     T   |
|           |
|   B       |
|       B   |
|   p       |
2 target(s) left
```

Initialization with example inputs

```
| T     T   |
|           |
|   B       |
|       B   |
|   p       |
2 target(s) left

Enter move: w

| T     T   |
|           |
|   B       |
|   p   B   |
|           |
2 target(s) left
```

In this example, the player is moved up by pressing "w" on the keyboard.