Outcome 1

Report

Class: 16SD

Name: Zhao Jichen

SCN: 187115469

Contents

1. Background Description

As a type of puzzle video game, Sokoban allows the player to play a role of a tiny man (a warehouse keeper) to push crates around in a warehouse, trying to get them to diamonds (storage locations). Sokoban was created in 1981 by Hiroyuki Imabayashi, and published in December 1982 by Thinking Rabbit, a software house based in Takarazuka, Japan.

The game is played on a board of squares, where each square is a floor or a wall. Some floor squares contain crates, and some floor squares are marked as diamonds. A tiny man is confined to the board, and may move horizontally or vertically onto empty squares (never through walls or crates). The tiny man can also move into a crate, which pushes it into the square beyond. Crates may not be pushed into other crates or walls, and they cannot be pulled. The number of crates is equal to the number of diamonds. The puzzle is solved when all crates are at diamonds.

2. Program Algorithm

2.1. Painting / Repainting a map

When the program starts to run, the map of Level 1 will be loaded and painted to the game area. When any element is required to change, the map will be repainted.

To repaint / repaint a map, first generate a map which shows the tiny man in real time. Then, generate an original map.

Get the position of the tiny man in the map. Paint elements according to the map (numbers recorded from 0 to 9). After that, clear undoing records.

2.2. Moving the Tiny Man

If the tiny man tries to move (up / down / left / right) a step by pressing a specified directive key (↑ / ↓ / ← / →), check one specified element around (above / below / on the left side of / on the right side of) the tiny man before moving.

- Situation 1: the wall

The tiny man stays still. Set a specified side (front / left / right / back) of the tiny man to the element at the position of the tiny man, and repaint the map.

- Situation 2: the ground inside or a diamond

  If the element at the position of the tiny man before moving is originally a diamond or a crate on a diamond, set a diamond to it.

  Otherwise, set the ground inside to it.

  (The above 2 paragraphs will be shortened to “restore a diamond or the ground inside after the tiny man moves”.)

  Set a specified side of the tiny man to the element around the tiny man before moving, and repaint the map.

  Mark the relevant action to a specified integer (10 / 20 / 30 / 40) and push it to the stack used as undoing records.

- Situation 3: a crate or a crate on a diamond

  If the element around the element around the tiny man before moving is originally a diamond, first restore a diamond or the ground inside after the tiny man moves, and then set a specified side of the tiny man to the element around the tiny man before moving and set a crate on a diamond to the element around the element around the tiny man before moving, and repaint the map. Mark the relevant action to a specified integer (11 / 21 / 31 / 41) and push it to the stack used as undoing records.

  Otherwise, if the element around the element around the tiny man before moving is originally the ground inside, first restore a diamond or the ground inside after the tiny man moves, and then set a specified side of the tiny man to the element around the tiny man before moving and set a crate to the element around the element around the tiny man before moving, and repaint the map. Mark the relevant action to a specified integer (11 / 21 / 31 / 41) and push it to the stack used as undoing records.

  Otherwise, the tiny man stays still. Set a specified side of the tiny man to the element at the position of the tiny man, and repaint the map.

2.3. Undoing a Step

The feature of this function is nearly the same as that of a last-in-first-out (LIFO) stack.

Therefore, the program uses class Stack from the package java.util to implement undoing a step. When the player moves the tiny man, a specified integer representing a kind of action will be pushed. When the player tries to undo a step, an integer will be popped, and action is performed according to the value popped.

- Situation 1: value popped is 10 / 20 / 30 / 40

  Restore a diamond or the ground inside after the tiny man moves.

  Set a specified side of the tiny man to the element at the position of the tiny man, and repaint the map.

- Situation 2: value popped is 11 / 21 / 31 / 41

  If the element at the position of the tiny man before moving is originally a diamond or a crate on a diamond, set a crate on a diamond to it.

  Otherwise, set a crate to it.

  If the element around the tiny man before moving is originally a crate or a crate on a diamond, set a crate to it.

  Otherwise, set the ground inside to it.

  Set a specified side of the tiny man to the element at the position of the tiny man, and repaint the map.

2.4. Changing a Level

- Situation 1: replay the current level

  Repaint the map.

- Situation 2: go to a specified level

  This includes last level, next level, Level 1, the final level, and level selected.

  First, set a specified integer from 1 to 5 to the variable “level”. Then, repaint the map.

2.5. Playing or Stopping BGM

As long as the program runs, the BGM Blue Danube will be played, and will not be stopped unless the player tells the program to do so.

To play or stop BGM, the program will first check if BGM is played. If so, then stop the player for a specified MIDI file, close it, and set “BGM OFF” to the text of the button controlling playing or stopping BGM.

Otherwise, try to get info of a MIDI file loaded to a sequence, obtain the default sequencer for playing back the MIDI sequence for BGM, and start to play it repeatedly until the player selects another BGM. During the process, if any exception occurs, show an error message. Set “BGM ON” to the text of the button.

2.6. Selecting BGM

When the player changes BGM selected, the program will first check if BGM is played. If so, then stop the player for the current MIDI file played, and close it.

Set the player’s selection to the MIDI file played. Then, try to get info of the specified MIDI file and load it to a sequence, obtain the default sequencer for playing back the MIDI sequence for BGM, and start to play it repeatedly until the player selects another BGM. During the process, if any exception occurs, show an error message. Set “BGM ON” to the text of the button.