CMSC 330: Organization of Programming Languages

Project 6 – Sliding Puzzle in Prolog

Sliding Puzzle

- Numbered tiles in a board
  - One empty space
- Move
  - Slide adjacent tile into space
- Continue until
  - All tiles are in order
  - Space in top left corner

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Puzzle Representation

- Represent puzzle as int list
- 2D array (in row-major order)
  - (0,0) (0,1) (0,2)
    (1,0) (1,1) (1,2)
    (2,0) (2,1) (2,2)
- 2D coordinates stored in 1D
  - (0,0) (0,1) (0,2) (1,0) (1,1) (1,2) (2,0) (2,1) (2,2)
- 1D positions
  - 0 1 2 3 4 5 6 7 8
- 2D (x,y) coordinate ↔1D position
  - X goes down, Y goes right (opposite normal behavior)

Puzzle Representation in Prolog

- Board is a list
  - [0,1,2,3] (0 = space, sorted = solved)
- Solution is a list of lists
  - [[1,0,2,3], [0,1,2,3]]
    - Adjacent boards result of a single move
    - No duplicate boards in solution
    - Boards start with original configuration & end at solved board

Project 6

- Implement utility functions in Prolog
  - Together can be used to solve puzzle
- Learn to use
  - Lists
  - Recursion

Project Files

- Your code
  - puzzle.pl
- Public tests
  - publicRecursion1.pl
  - publicRecursion2.pl
  - publicPuzzle.pl
  - publicSolve.pl
- Utility files
  - goTest.pl - test driver for all public tests

Public Tests

- Contain following test goals
  - get_val_public
  - get_vals_public
  - set_n_public
  - list_swap_public
  - index_public
  - move_pos_public
  - make_move_public
  - make_moves_public
  - single_move_public
  - solve_board_public

Differences From Project 3

- Utilize backtracking in Prolog
  - Return single answer, instead of list of answers
  - Request additional answers by typing `;`
- Tests will collect all answers, using Prolog utils
  - Findall – list of all answers
  - Setof – order list of all answers (duplicates removed)

Using Prolog From Terminal / Shell

- Go to directory p6 (from p6.zip) containing proj files
  - E.g., cd c:Users\myname\Desktop\p6
- Start Prolog interpreter
  - Type **`swipl`**
- Load Prolog code
  - Type **`['puzzle.pl'].`**
  - Type **`['publicRecursion1.pl'].`**
- Run public tests
  - Type **`get_val_public.`**
  - Etc…

Using Prolog From Terminal / Shell

- To run all public tests
  - Type **`['goTest.pl'].`**
  - Type **`run.`**
- To modify your code
  - Edit puzzle.pl
  - Type **`make.`**

Using Prolog From Windows

- Open folder p6 (from p6.zip) containing project files
- Start Prolog interpreter
  - Right click goTest.pl, select Open With -> swipl-win.exe
  - Terminal window opens running Prolog interpreter
    - goTest.pl loaded at same time
- To run all public tests
  - Type **`run.`**
- To modify your code
  - Edit puzzle.pl
  - Type **`make.`**