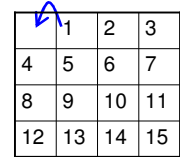


CMSC 330: Organization of Programming Languages

Project 3 – Sliding Puzzle

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

CMSC 330

2

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

CMSC 330

3

Puzzle Representation in OCaml

- ▶ int list = board
 - [0;1;2;3] (0 = space, sorted = solved)
- ▶ int list list = list of boards
 - [[1;0;2;3]; [0;1;2;3]]
 - May indicate a solution if
 - Adjacent boards result of a single move
 - Boards start with original configuration & end at solved board
- ▶ int list list list = list of solutions
 - [[[1;0;2;3]; [0;1;2;3]] ; [[2;1;0;3]; [0;1;2;3]]]

CMSC 330

4

Project 3

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

CMSC 330

5

Project 3

- ▶ Functions use tuples
 - index (x, v) 'a list * 'a -> int
- ▶ In general, assume legal inputs
 - Up to you to ensure used with legal inputs in solver

CMSC 330

6

Using OCaml

- ▶ Run interpreter in shell (Linux, Apple)
 - Go to directory p3 (from p3.zip) containing project files
 - E.g., `cd c:\Desktop\p3`
 - Type `ocaml puzzle.ml`
 - Compiles & runs code in puzzle.ml
- ▶ Run OCamlWinPlus executable in Windows
 - Open OCamlWinPlus in folder p3 containing project files
 - Type `#use "puzzle.ml"` in OCamlWinPlus window
 - Compiles & runs code in puzzle.ml