Assignment 4

Team Members:

1. Dhrumil Amish Shah

Date:

2021-03-13

—

Subject:

Software Development Concepts

—

Professor:

Matthew Amy

Test Cases

⇨ Input Validation Cases

1.) public boolean loadPuzzle(BufferedReader stream)

- Send stream null.
  - Returns false.
- Send invalid puzzle size. (Example: In words like four, other than int type like float, char, String etc...) – First line in the stream
  - Returns false.
- Send board of size not equal to entered size. (Valid: Puzzle board size = size * size) (Invalid example: Puzzle of size 5 and board of length 4)
  - Returns false.
- Send invalid constraint format. (Valid: group<space>outcome<space>operator)
  - Returns false.
- Send space in the input format anywhere.
  - Returns false.
- Send size less than 2.
  - Returns false.

2.) public boolean validate()

- Send size less than 2.
  - Returns false.
- Puzzle not of size * size.
  - Returns false.
- Every grouping is a connected set of cells.
  - Returns true.
- Every grouping with the '=' operator has exactly one cell
  - Returns true.
- Every grouping with '-' or '/' operator has exactly two cells.
  - Returns true.
- Every grouping with '+' or '*' operator has at least two cells.
  - Returns true.
- All groups are present in the constraints.
  - Returns true.
- Any group missing in the constraints.
  - Returns false.
- Extra group present in the constraints.
  - Returns false.
- Spaces between the groupings passed.
  - Returns false.

3.) public boolean solve()

- No input validation cases.

4.) public String print()

- No input validation cases.

5.) public int choices()

- No input validation cases.

⇨ Boundary Cases

1.) public boolean loadPuzzle(BufferedReader stream)

- Send long stream of data.
  - Returns true if valid otherwise false.
- Send short stream of data.
  - Returns true if valid otherwise false.
- Send puzzle size 2.
  - Returns true if stream is valid otherwise false.

2.) public boolean validate()

- No boundary cases.

3.) public boolean solve()

- No boundary cases.

4.) public String print()

- No boundary cases.

5.) public int choices()

- No boundary cases.

⇨ Control Flow Cases

1.) public boolean loadPuzzle(BufferedReader stream)

- No control flow test cases.

2.) public boolean validate()

- No control flow test cases.

3.) public boolean solve()

- Solve when multiple solution exists for a puzzle.
  - Returns true.
- Solve a solvable puzzle.
  - Returns true.
- Solve an unsolvable puzzle.
  - Returns false.
- Solve a puzzle with few numbers of operator constraints.
  - Returns true if solution is found otherwise false.
- Solve a puzzle with all the operator constraints.
  - Returns true if solution is found otherwise false.

4.) public String print()

- Print partially solved puzzle.
  - Returns a string of partially solved puzzle.
- Print an unsolved puzzle.

    - Returns a string of unsolved puzzle.
- Print a solved puzzle.
    - Returns a string of solved puzzle.

5.) public int choices()

- Puzzle not solved.
    - Returns 0.
- Puzzle solved.
    - Returns a value > 0.

⇨ **Data Flow Cases**

⇨ **Note:** I am calling the validate() method inside the solve() method. If validate() passes then only I will solve() the puzzle.

1.) public boolean loadPuzzle(BufferedReader stream)

- Call multiple times on the same puzzle.
    - Returns true if the puzzle is loaded otherwise false.

2.) public boolean validate()

- Call before calling loadPuzzle.
    - Returns false.
- Call multiple times on the same puzzle after loaded.
    - Returns true if the puzzle is valid otherwise false.

3.) public boolean solve()

- Call before calling loadPuzzle.
    - Returns false.
- Call multiple times on the same puzzle after loaded.
    - Returns true if a solution is found otherwise false.

4.) public String print()

- Call before calling loadPuzzle().
    - Returns false.
- Call before calling validate method.
    - Returns the puzzle grouping.
- Call before calling solve method.
    - Returns the puzzle grouping.
- Call after calling solve method.
    - Returns solved puzzle if a solution is found otherwise combined grouping solution.
- Call multiple times on the same puzzle after loaded.
    - Returns false.

5.) public int choices()

- Call before calling loadPuzzle.
    - Returns 0.
- Call before calling validate method.
    - Returns 0.

- Call before calling solve method.
    - Returns 0.
- Call after calling solve method.
    - Returns a number > 0 whether a solution is found or not.
- Call choices multiple time.
    - Returns the value of choices if solved otherwise false.