Проект по ООП. Реализация судоку. Ильин Семён ФБКИ ИГУ.

ТЗ:

1. **Разработайте консольный вариант игры «Судоку».**

Правила:

Игровое поле представляет собой квадрат Любой размерности, разделённый на меньшие квадраты. В некоторых точках уже в начале игры стоят некоторые числа, называемые *подсказками*. От игрока требуется заполнить свободные клетки , так чтобы в каждой строке, в каждом столбце и в каждом малом квадрате каждая цифра встречалась бы только один раз. Сложность судоку зависит от количества изначально заполненных клеток и от методов, которые нужно применять для её решения. Самые простые решаются дедуктивно: всегда есть хотя бы одна клетка, куда подходит только одно число.

Анализ задачи

Архитектурные решения
Структура программного комплекса

Система организована в виде пяти ключевых модулей:

Модуль Board
Ответственен за хранение состояния игрового поля и проверку корректности вводимых значений. Использует композитную структуру из объектов Cell и Block для оптимизации проверок уникальности значений.

Модуль GameEngine
Реализует бизнес-логику игрового процесса, включая управление историей ходов и определение момента завершения игры.

Модуль PuzzleGenerator
Обеспечивает генерацию начальных условий задачи с использованием алгоритма обратного отслеживания (backtracking) и последующего удаления части чисел в зависимости от выбранного уровня сложности.

Модуль ConsoleUI
Обрабатывает пользовательский ввод через командную строку и

предоставляет текстовую визуализацию игрового состояния.а

Модуль Main
Запускает программу

```
project/
│
├── cpp/
│   ├── board.cpp
│   ├── consoleUI.cpp
│   ├── difficult.cpp
│   └── gameEngine.cpp
│
├── h/
│   ├── board.h
│   ├── consoleUI.h
│   ├── difficult.h
│   └── gameEngine.h
│
├── main.cpp
```

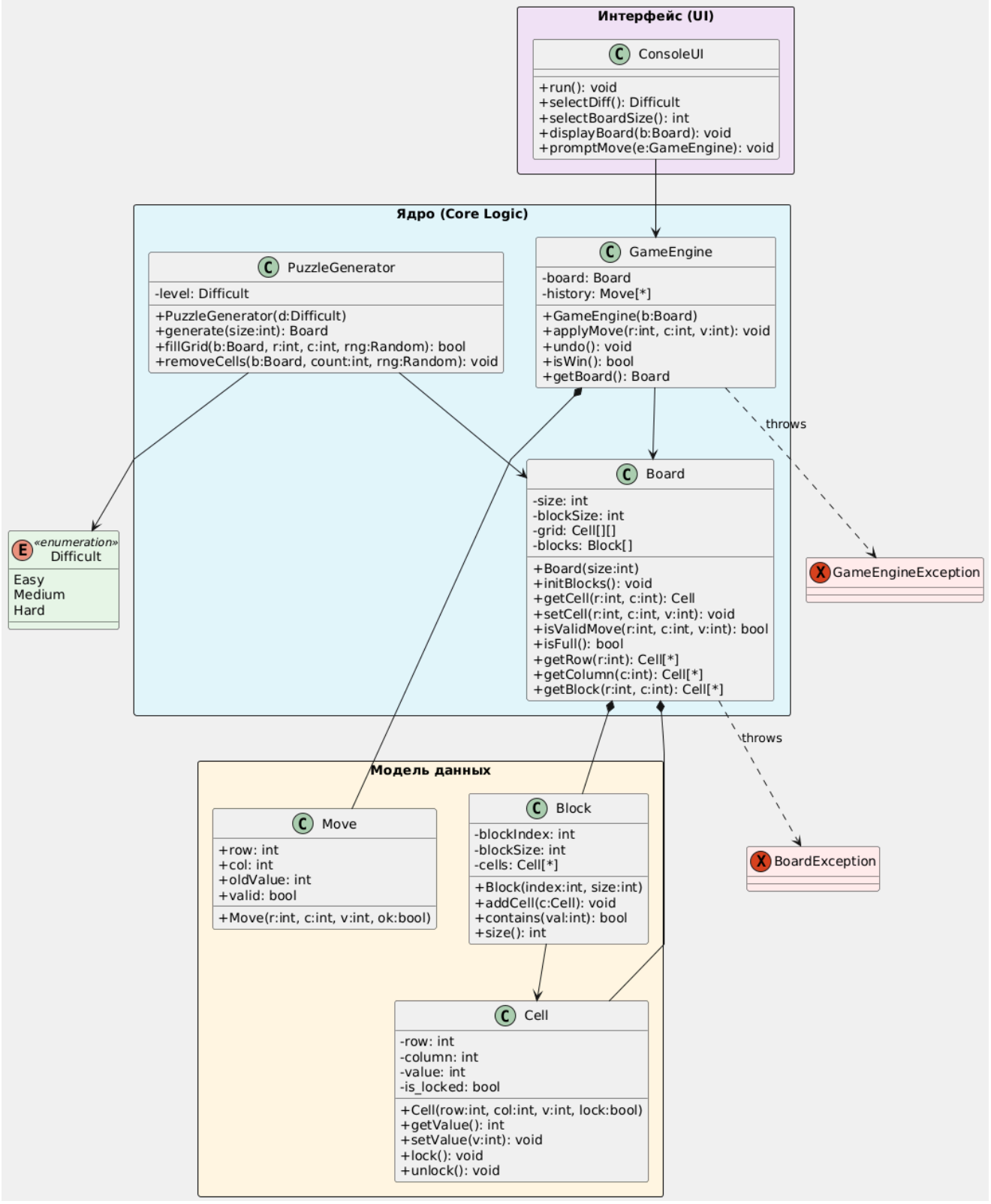
Механизм генерации головоломок

Алгоритм генерации включает два этапа: создание полностью заполненного корректного поля и последовательное удаление чисел с сохранением единственности решения. Уровень сложности регулируется процентом удаляемых элементов (20-60%).
Система проверки ходов

Валидация вводимых значений осуществляется через трёхуровневую проверку: анализ уникальности в текущей строке, столбце и блоке 3×3.
Поддержка различных размеров поля

Архитектура позволяет использовать поля произвольного размера при условии, что размер является полным квадратом. Вычисление границ блоков выполняется через определение целочисленного квадратного корня от общего размера поля.
Технические требования

3. UML-диаграмма



Основные модули и ключевые функции

Board (игровое поле)

```
Board(int size) - конструктор, создаёт пустую сетку и инициализирует блоки

bool isValidMove(int row, int col, int val) - проверяет, можно ли поставить val в ячейку

void setCell(int row, int col, int val) - устанавливает значение в ячейку

bool isFull() const - проверяет, заполнено ли поле полностью
```

GameEngine (игровая логика)

```
GameEngine(int boardSize) / GameEngine(const Board& initialBoard) - создание движка с пустым или заданным полем

void applyMove(int row, int col, int value) - выполняет ход, сохраняет в историю

void undo() - отменяет последний ход

bool is_win() const - проверяет условие выигрыша (поле заполнено)
```

PuzzleGenerator (генератор головоломок)

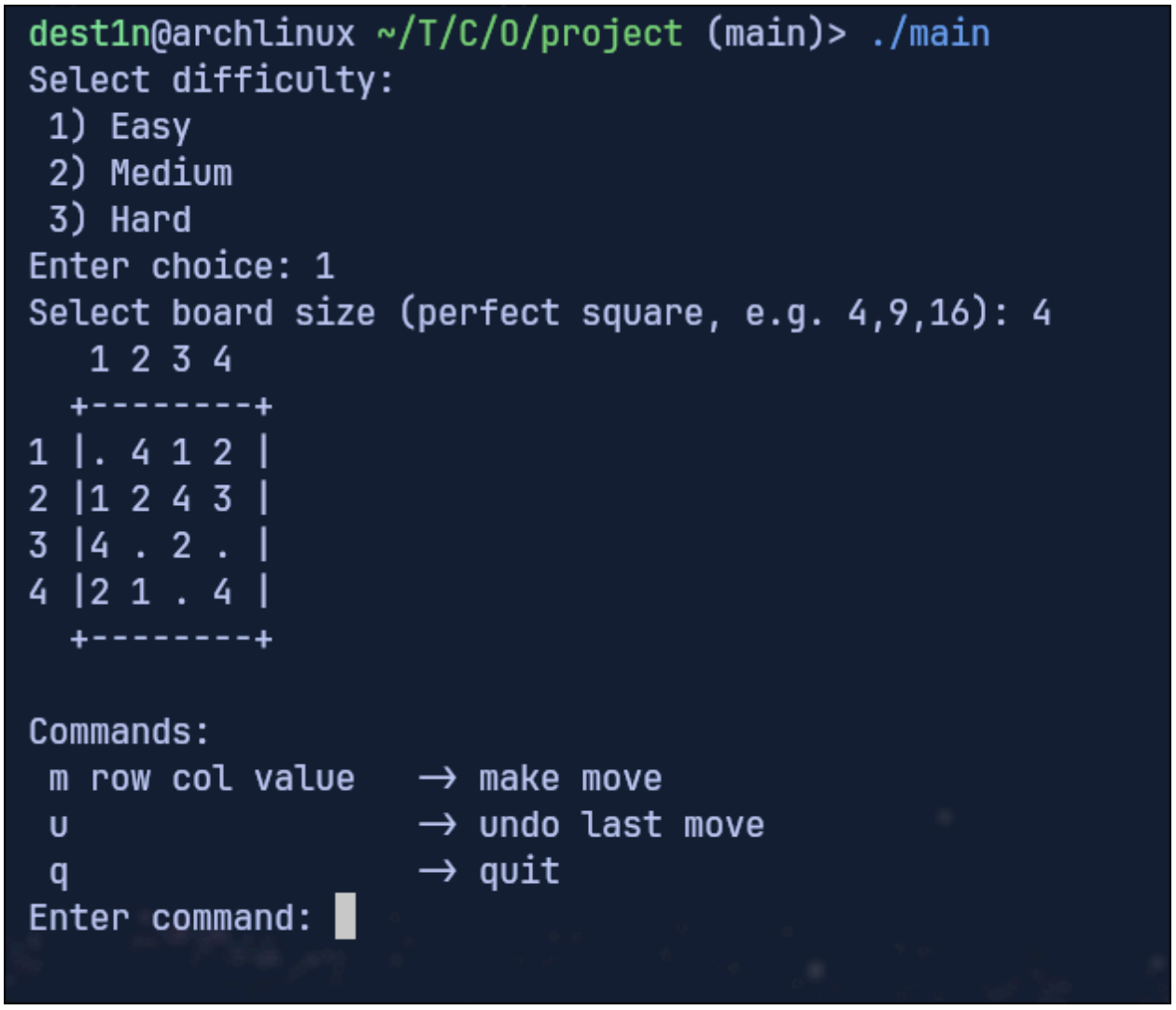
```
Board generate(int size) - создаёт полностью заполненное поле и удаляет числа по сложности

bool fillGrid(Board& board, int row, int col, mt19937& rng) - рекурсивная функция обратного отсчёта (backtracking)

void removeCells(Board& board, int removeCount, mt19937& rng) - случайно удаляет заданное число ячеек
```

ConsoleUI (пользовательский интерфейс)

```
void run() - главный цикл программы: выбор сложности, размера, генерация и игра

void displayBoard(const Board& board) - текстовый вывод текущего состояния

void promptMove(GameEngine& engine) - ввод команд: m row col value, u, q
```

4. Тесты

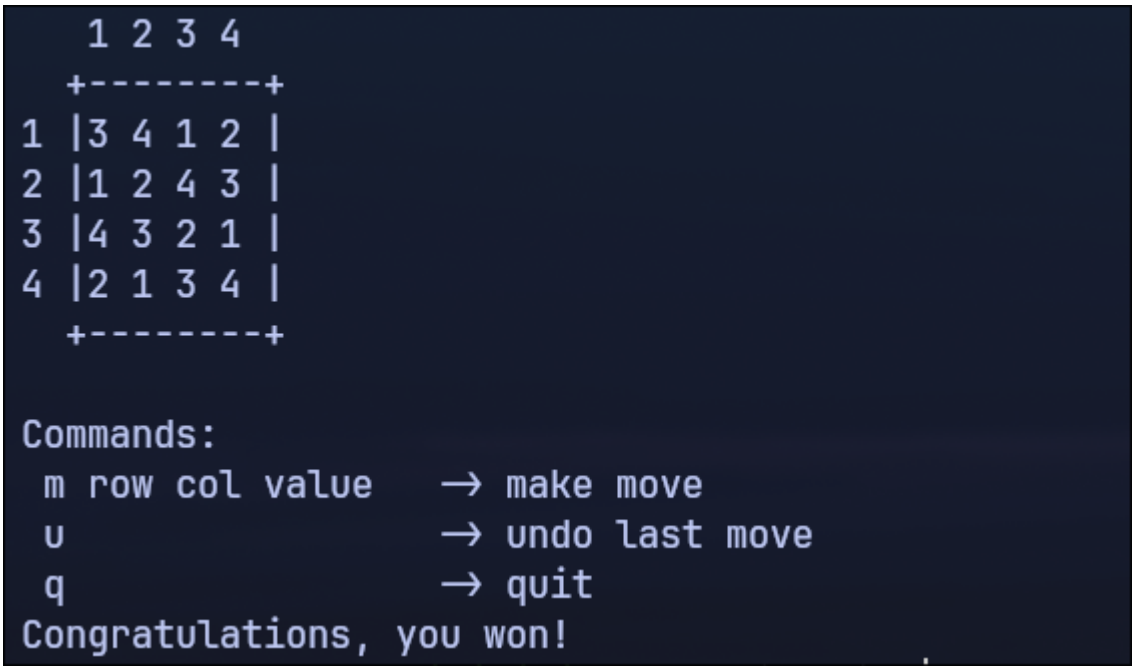
1. 4X4 EASY


dest1n@archlinux ~/T/C/O/project (main)> ./main
Select difficulty:
1) Easy
2) Medium
3) Hard
Enter choice: 1
Select board size (perfect square, e.g. 4,9,16): 4
1 2 3 4
+--------+
1 |. 4 1 2 |
2 |1 2 4 3 |
3 |4 . 2 . |
4 |2 1 . 4 |
+--------+
Commands:
m row col value → make move
u → undo last move
q → quit
Enter command:

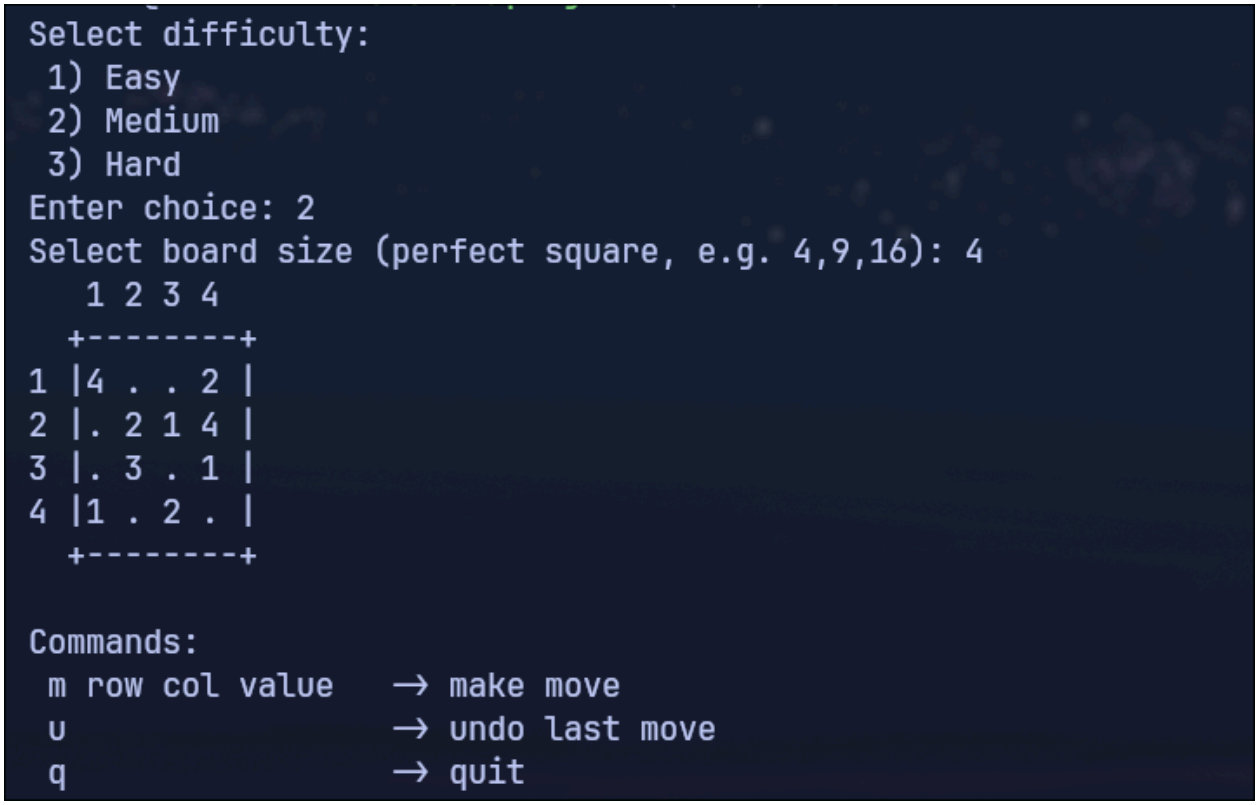
C++

```
Enter command: m 1 1 3
   1 2 3 4
  +--------+
1 |3 4 1 2 |
2 |1 2 4 3 |
3 |4 . 2 . |
4 |2 1 . 4 |
  +--------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
Enter command: m 3 2 3
   1 2 3 4
  +--------+
1 |3 4 1 2 |
2 |1 2 4 3 |
3 |4 3 2 . |
4 |2 1 . 4 |
  +--------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
Enter command: m 4 3 3
   1 2 3 4
  +--------+
1 |3 4 1 2 |
2 |1 2 4 3 |
3 |4 3 2 . |
4 |2 1 3 4 |
  +--------+

Commands:
 m row col value   → make move
 u                 → undo last move
```

```
 q                  → quit
Enter command: m 3 4 1
```



2. MEDIUM 4X4



```
   1 2 3 4
  +--------+
1 |4 1 3 2 |
2 |3 2 1 4 |
3 |2 3 4 1 |
4 |1 4 2 3 |
  +--------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
Congratulations, you won!
dest1n@archlinux ~/T/C/O/project (main)>
```

3. HARD 4X4

```
Select difficulty:
 1) Easy
 2) Medium
 3) Hard
Enter choice: 3
Select board size (perfect square, e.g. 4,9,16): 4
   1 2 3 4
  +--------+
1 |. . . 2 |
2 |3 . . . |
3 |. . . . |
4 |1 4 2 3 |
  +--------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
Enter command:
```

```
   1 2 3 4
  +-------+
1 |4 1 3 2 |
2 |3 2 4 1 |
3 |2 3 1 4 |
4 |1 4 2 3 |
  +-------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
Congratulations, you won!
```

4. Easy 9X9

```
   1 2 3 4 5 6 7 8 9
  +-----------------+
1 |. . 2 4 8 9 6 7 3 |
2 |6 4 7 2 1 3 8 9 5 |
3 |9 . 3 5 6 7 1 2 4 |
4 |7 . 4 . 2 . . 5 8 |
5 |2 5 1 8 . 4 7 6 9 |
6 |. . . 9 7 . 4 1 2 |
7 |4 2 . 3 . 1 5 8 7 |
8 |. 3 9 7 5 . 2 4 6 |
9 |8 7 5 6 4 . 9 3 1 |
  +-----------------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
```

```
Move successful!
   1 2 3 4 5 6 7 8 9
  +-------------------+
1 |5 1 2 4 8 9 6 7 3 |
2 |6 4 7 2 1 3 8 9 5 |
3 |9 8 3 5 6 7 1 2 4 |
4 |7 9 4 1 2 6 3 5 8 |
5 |2 5 1 8 3 4 7 6 9 |
6 |3 6 8 9 7 5 4 1 2 |
7 |4 2 6 3 9 1 5 8 7 |
8 |1 3 9 7 5 8 2 4 6 |
9 |8 7 5 6 4 2 9 3 1 |
  +-------------------+

Commands:
 m row col value   → make move
 u                 → undo last move
 q                 → quit
Congratulations, you won!
```

5. Проверка запуска 25x25 и 16x16

```
Select board size (perfect square, e.g. 4,9,16): 16
   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
  +--------------------------------+
1 |. @ 3 9 6 > 2 ; : 7 < . 5 8 4 . |
2 |4 . 7 ? 9 : 5 = 2 3 6 @ 1 . > < |
3 |. . 1 < ? . 4 8 > . = 9 : 3 6 @ |
4 |5 > : 6 < 3 1 @ . . ? 4 = 7 2 . |
5 |8 4 ; 1 5 @ . . 9 2 . . . . = 3 |
6 |? : < . 1 8 = . @ 6 ; 3 . . 5 4 |
7 |3 = 5 @ : ? . . 8 4 1 7 > 6 9 ; |
8 |. 6 9 2 . 4 . > ? . 5 < 8 . @ . |
9 |< 1 4 . 7 . . . = ; 8 5 @ > : 2 |
10 |6 7 = ; > 5 3 < 1 @ : 2 4 9 . . |
11 |. . @ 5 = 1 8 . . > 9 . ; < 7 6 |
12 |9 ? > 8 2 ; @ : . . 7 6 3 = 1 . |
13 |@ 5 ? : 8 = 6 1 7 9 3 ; 2 4 < > |
14 |. < 6 . @ 9 > 7 5 . . 8 ? : 3 = |
15 |. 3 2 = . < : 5 6 ? 4 > 9 @ 8 7 |
16 ▷ 9 8 . 4 2 ? 3 < : @ = 6 5 . 1 |
  +--------------------------------+
```


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
1 |; > E I G 7 . . = . H A 4 5 . 8 1 B 9 . 2 C ? 3 D |
2 |. 8 @ 5 F > 9 ? H 4 2 ; E C 3 . . D < 7 . A 1 : B |
3 |1 A < D C . : . ; . = 7 9 F . 4 . > 2 3 @ 5 G E H |
4 |4 7 H B 2 . C 3 E 1 ? 8 : @ D A I . 5 ; 9 6 = > < |
5 ⊨ . 9 . ? . 5 . < . 1 I G . > C . @ E . 7 . 4 F . |
6 |8 2 B ? . . ; A C = : 1 < 9 7 > 4 I G 6 3 D . H E |
7 |C < F ; 7 H 4 > . ? E 6 2 . . B 3 : @ D = 9 5 I G |
8 |A I G . E 2 3 < 9 B ; F 5 D @ 1 H = 8 ? C > . 7 : |
9 |3 D = @ 9 8 6 E : 5 4 . H > I 2 ; A . C F . < 1 ? |
10 |: 6 1 H > I G . 7 @ C . . 3 = 5 9 E F < A 8 . 4 2 |
11 |2 1 ; G . 4 E 7 . 9 @ ? F : 5 . 8 3 6 = B H A < C |
12 |. 3 C = . . F : . G D 2 7 1 9 I E . H . ; . > 5 . |
13 |5 . > 8 D A . 1 B 3 6 E ; < . F 7 . C @ : G I ? . |
14 |E . A . 6 . H = D . > 4 I G B ; 2 . : 1 . 3 F 9 7 |
15 |. F 4 : H ? < I @ ; . . . . . > 5 . G . 2 E 6 1 |
16 |H . 5 . 4 . 7 @ ? 2 < 9 A . . : B 1 I > G . 8 D F |
17 |I G 8 C : = B 4 A D . . . 6 2 < . H ; E 1 ? 9 @ 5 |
18 |D ; 2 > @ E 1 G I < B : 8 . F 6 5 9 3 4 H = 7 . A |
19 |< B 3 9 . C > ; . . 5 D @ E 1 = G 7 ? 8 4 I : 2 6 |
20 |F . ? 7 1 9 8 . 6 : I H = . G @ A C . 2 > < B ; . |
21 |9 H I 2 . < ? . 3 > . = D 7 . . . . 1 5 . 4 C . ; |
22 |@ C : A 3 ; I 9 . . G 5 1 H ? . < 6 4 . E F . = > |
23 |. . 6 . . @ . . 4 7 9 > C 2 E ? D 8 . F < : . B . |
24 |. 4 D E < . . C 5 . . @ 6 A ; 3 : 2 > I ? 7 H G 9 |
25 ▷ ? 7 F . : D . G E 3 < B I 4 . . ; A 9 5 1 2 8 . |