**Jewel match**

This game is similar to other matching games like Bejewelled. The main points to consider are:

* The player is presented with a 10 by 10 grid of jewels.
* The player is able to choose a jewel to swap with its left or right neighbour.
* The player starts with 3 moves and loses one every swap.
* A jewel is chosen by its x and y coordinates.
* As a result, the grid has number guides to aid the player in getting the correct coordinates.
* When a jewel is swapped, the game will check to see if there is a match.
* A match consists of 3 or more consecutive jewels in a row, horizontal or vertical.
* All matching jewels will be removed from the grid.
* If jewels end up with space below them, then they’ll be made to fall into place.
* Once all jewels have been dropped into position, then any empty space will be filled with jewels.
* At this point the game will check for matches again.
* If further matches are found, then a combo meter will be incremented by one.
* Once no matches are present on the grid, then the player will be able to swap another jewel.
* Moves and score, modified by combo, will be awarded depending on the number of matched jewels.
* The game will end when the player runs out of moves.

**Function list**

**create\_playfield**

Creates a grid with a random arrangement of jewels.

**numbers\_x**

Outputs x coordinate guides, helper function for display\_playfield.

**display\_playfield**

Outputs the playfield with number guides.

**remove\_jewels**

Removes jewels from the playfield from coordinates specified in a list.

**drop\_jewels**

Make ‘floating’ jewels fall into place on top of other jewels.

**fill\_empty**

Fills empty spaces in the playfield with random jewels.

**swap\_jewels**

Allows the player to swap a jewel with its neighbour.

**find\_matches**

Attempts to find 3 or more consecutive rows of jewels horizontally and vertically.

**prepare\_field**

Do remove\_jewels, drop\_jewels and fill\_empty one after the other.

**main**

Main game loop.

**Script to script changes**

**00**

Program skeleton

**01**

Create and display playfield

**02**

Let the player select a jewel

**03**

Let the player swap a jewel with its neighbour

**04**

Find horizontal matches in the playfield

**05**

Find vertical matches in the playfield

**06**

Remove jewels found by the matching function

**07**

Make ‘floating’ jewels fall into place

**08**

Fill empty spaces in the playfield

**09**

Create function to combine remove, drop and fill jewels functions

**10**

Update playfield display function by adding number guides

**11**

Outline structure of main game loop

**12**

Start game with a combo-less playfield

**13**

Implement core game logic

**14**

Complete game with score and move replenishment