My idea Yunhao Chen, 15 November 2023

Username: Lozlant

Link: https://github.com/Lozlant/Elemental\_Pinball

|  |
| --- |
| **Elemental Pinball**  The game consists of two menus: the start screen and the settlement screen. There are three attributes for the pinball: Fire, Thunder, and Ice. Players can change the board by pressing number keys, and when the ball hits the board, it adopts the board's attribute.  When the ball hits a block with no attribute, it imparts the corresponding attribute to the block. If it hits a block with an attribute, an elemental reaction occurs:   * Fire + Thunder: Explosion (with special effects) destroys blocks in a cross-shaped range. * Fire + Ice: Melting (with special effects) turns surrounding adjacent blocks into ice elements. * Thunder + Ice: Superconductivity (with special effects) destroys blocks in a horizontal row.   After achieving a combo of 30, a new ball splits. Settlement occurs when all balls disappear or after a certain duration.  There are three levels (attempting to generate levels randomly), and the highest record for each level is recorded.  In the Setting Menu, players can end the game, return to the main menu, or view controls.    Pseudocode   1. Create a canvas and initialize values. 2. Create a class for the ball and a class for the block group. 3. For each frame: a) Fill the background. b) If the game has just started, display the start screen. c) If the game is over (all blocks are cleared or countdown reaches zero), display the settlement screen. d) If the game is in the setting state, display the setting screen. e) In-game:    1. If there are no balls on the field, allow clicking to launch a new ball.    2. If there are balls on the field, calculate ball movement, settle collisions, check if the ball is out of bounds, and inspect combo and ball splitting.    3. Settle effects. 4. Key events: a) AD: Move the board. b) In the absence of a ball, spacebar: Launch a new ball. c) Number keys: Change the board's attribute. 5. Mouse events: a) Setting button. b) Start button. c) Return to the main menu button. d) Exit button. |

Where will the inventory skills be demonstrated? List every one to be sure you’ve included them.

|  |  |  |  |
| --- | --- | --- | --- |
| Shape  1. line, ellipse, rect, triangle, quad, arc, curve (create graphic elements)  2. fill, stroke, strokeWeight, noFill, noStroke, color (create graphic elements)  3. Modes: CORNER, CORNERS, CENTER, RADIUS (create graphic elements)  System  4. setup(), draw() (main program)  5. background() (overwrite content in a new frame), random(), noise()  6. constrain(), dist() (limit the movement range of the board)  7. keyPressed(), keyReleased(), keyPressed(), mousePressed(), mousePressed() (mouse clicks on menu buttons and keyboard control of the board)  8. increment operators: ++, +=, --, -=, \*=, /= (data processing, such as score increase, ball movement)  9. declare and use a local variable (used in many places for local data processing)  10. declare and use a global variable (balls, boards, score, countdown, boolean variables indicating game phase and status, etc., are all global variables)  Debugger  11. println(), stop() (used in debugging, typically commented out later)  Control flow  12. conditional statements: if, else if, else (determine game phase, collision, etc.)  13. Boolean expressions: ==, >=, <=, >, <, != (most conditions are determined by these symbols)  14. Logical operators: &&, || (calculate game control conditions)  15. switch statement (used in specific situations)  Loops  16. for loop, while loop (monitor collision, calculate graphics)  17. A nested loop (monitor collision)  18. break() (exit after successful collision detection)  19. What’s the difference between a for loop and a while loop?  Function  20. Declare & call a function with no parameters and no return type (related to drawing functions)  21. Declare & call a function with a return type (collision detection returns)  22. What’s the difference between parameters and arguments?  23. Pass by copy (value): declare and use a function that takes int, float, char, etc., as an argument (passing new values to return the highest score)  24. Pass by reference (objects): declare and use a function that takes an object as an argument (passing balls and boards for collision detection)  Classes/Objects  25. What’s the difference between a class and an object?  26. What is a constructor function? What does it do and when?  27. Why should each class have its own tab in Processing?  28. Write a class with a constructor function (ball, board, block group)  29. Use the keyword new to instantiate an object (ball, board, block group)  30. Write a constructor function with parameters (ball, board, block group)  Lists  31. What’s the difference between an array and an ArrayList?  32. Why would you want to go through a list backward, decrementing the index?  33. Initialize and populate an array (group of balls to be used)  34. Initialize and populate an ArrayList (split balls, block group)  35. Manage a set of objects with an array or ArrayList (balls, block group)  36. Use an ArrayList method: size(), get(), remove(), contains() (when updating balls and block groups)  Vector  37. When should you use PVector instead of float variables?  38. Use the PVector class (record ball coordinates, ball velocity)  39. Do some basic physics: use position, velocity, and acceleration (due to gravity) vectors (record ball coordinates, ball velocity)  40. Find the direction and distance between two points (collision detection between ball and board)  41. What is a normalized vector, why is it useful?  Other  42. Use a timer (countdown)  43. Switch between “game states” (e.g., grounded/jumping) using conditional statements  44. Make a button or toggle switch with a roll-over highlight (color or size change)  45. Do animation with images (spritesheet or individual files)  46. Use collision detection between objects  47. Use the Game Control Plus controller library to get joystick or gamepad input | | | |
| **Milestone 1** | **Milestone 2** | **Milestone 3** | **Milestone 4** |
| Complete Design Sketch  Object creation and layout of the game interface  Build the main program of the game  Generate the block group  Complete the code for ball movement  Keyboard control  Collision and destruction of blocks  Ball splitting | Countdown  Write elemental reactions  Artistic refinement of game objects  Create operation effects, button effects  Write countdown functionality  Design game phases  Complete game | Design menu  Mouse control  Bind Game Control Plus  Enhance gameplay |  |
| * 1. setup(), draw() | * 1. conditional statements: if, else if, else | 3.1 switch statement |  |
| * 1. background(), random(), noise() | * 1. Initialize and populate an array | 3.2 Pass by copy (value): declare and use a function that takes int, float, char, etc as an argument |  |
| * 1. constrain(), dist() | * 1. Use a timer | 3.3 Do animation with images |  |
| * 1. keyPressed(), keyReleased(), keyPressed(), mousePressed(), mousePressed() | * 1. Switch between “game states” (eg grounded/jumping) using conditional statements | 3.4 Use the Game Control Plus controller library to get joystick or gamepad input |  |
| 1.5 increment operators: ++, +=, --, -=, \*=, /= | 2.5 Make a button or toggle switch with a roll-over highlight |  |  |
| * 1. declare and use a local variable | * 1. line, ellipse, rect, triangle, quad, arc, curve |  |  |
| * 1. declare and use a global variable | 2.7 fill, stroke, strokeWeight, noFill, noStroke, color |  |  |
| 1.8 Boolean expressions: ==, >=, <=, >, <, != | 2.8 Modes: CORNER, CORNERS, CENTER, RADIUS |  |  |
| * 1. Logical operators: &&, || | 2.9 println(), stop() |  |  |
| 1.10 for loop, while loop | 2.10 Answer questions  What’s the difference between a for loop and a while loop?  What’s the difference between parameters and arguments?  What’s the difference between a class and an object?  What is a constructor function? What does it do and when?  Why should each class have its own tab in Processing?  What’s the difference between an array and an ArrayList?  Why would you want to go through a list backwards, decrementing the index?  When should you use PVector instead of float variables?  What is a normalized vector, why is it useful? |  |  |
| * 1. A nested loop |  |  |  |
| * 1. break() |  |  |  |
| 1.13 Declare & call a function with no parameters and no return type |  | **Y** |  |
| * 1. Declare & call a function with a return type |  |  |  |
| * 1. Pass by reference (objects): declare and use a function that takes an object as an argument |  |  |  |
| * 1. Write a class with a constructor function   2. Use the keyword new to instantiate an object   3. Write a constructor function with parameters |  |  |  |
| * 1. Initialize and populate an ArrayList   2. Manage a set of objects with an array or ArrayList   3. Use an ArrayList method: size(), get(), remove(), contains() |  |  |  |
| * 1. Use the PVector class   2. Do some basic physics: use position, velocity, and acceleration (due to gravity) vectors   3. Find the direction and distance between two points |  |  |  |
| 1.25 Use collision detection between objects |  |  |  |