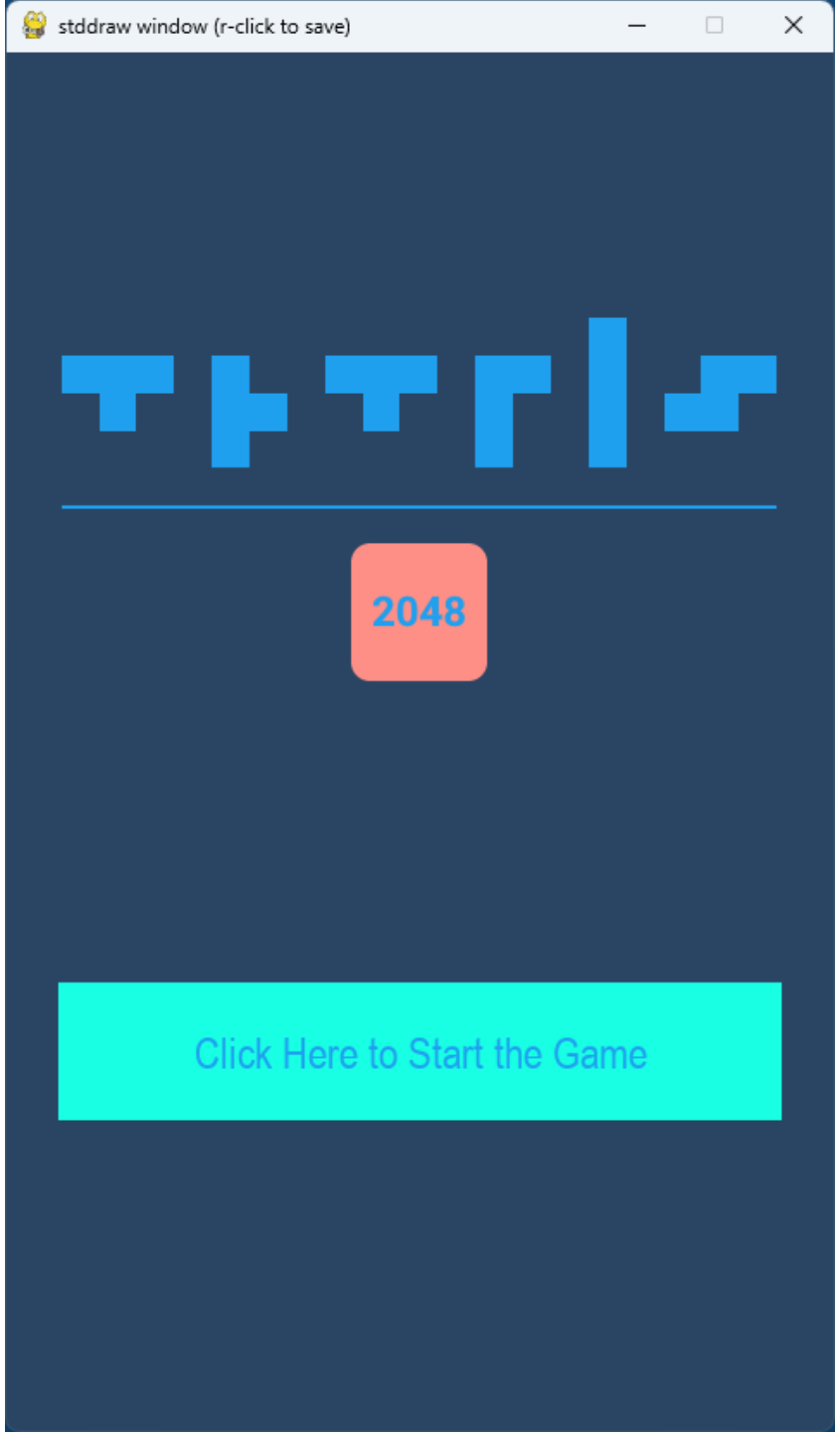
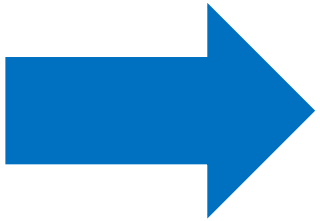
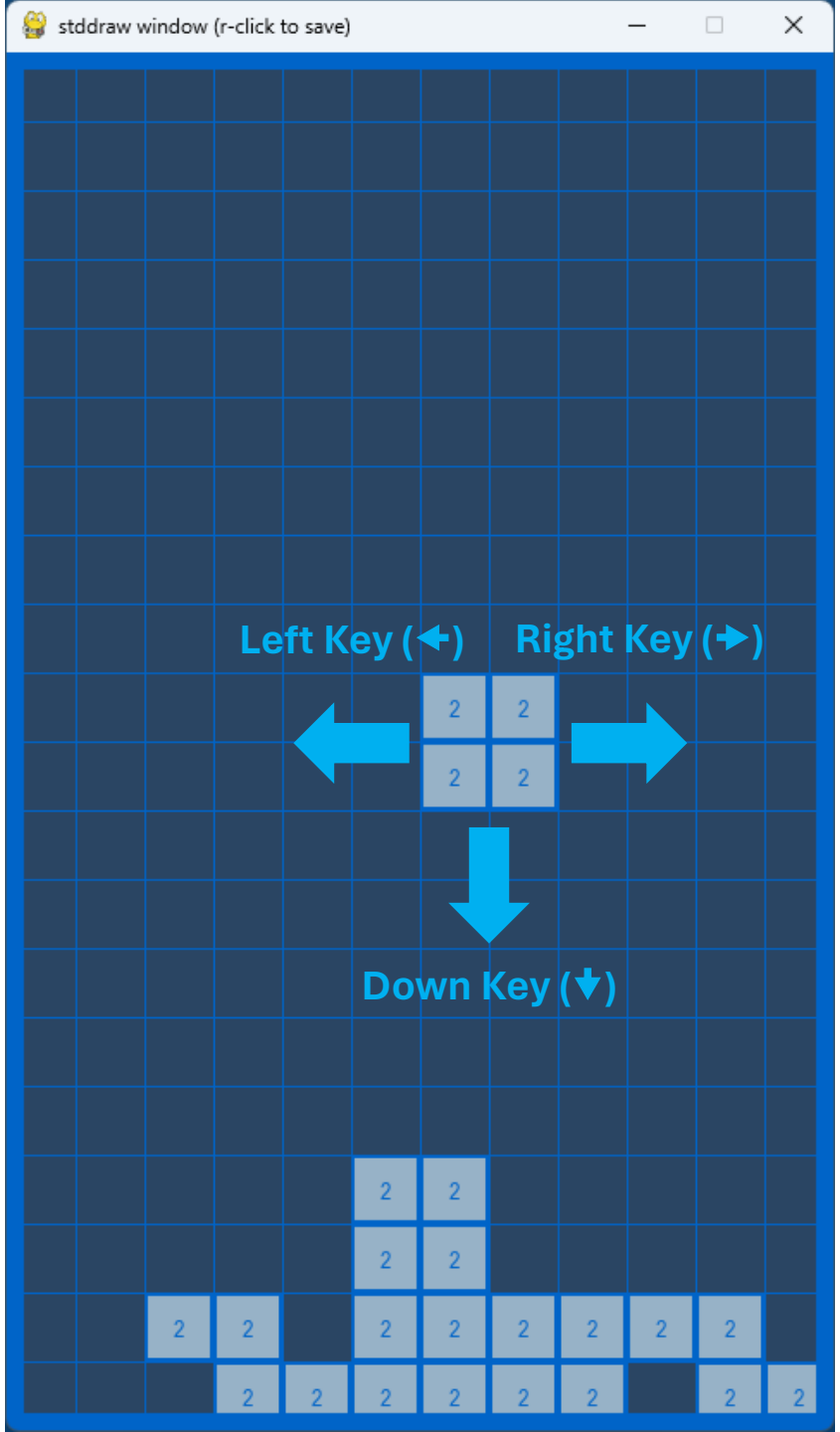
Tetris 2048 Base Code

Tetris 2048 Base Code

- Tetris 2048 base code consists of the following contents:
  - lib/stddraw.py which contains the stddraw module with some modifications
  - lib/color.py which contains the Color class that can be used with the stddraw module
  - lib/picture.py which contains the Picture class that can be used with the stddraw module
  - point.py which contains the Point class for modeling a point as a location (x, y) in 2D space
  - tile.py which contains a simple Tile class (This class can represent numbered tiles with only 2 as the number.)
  - tetromino.py which contains a Tetromino class (This class can represent only 3 out of 7 different tetrominoes and methods for rotating tetrominoes are not included.)
  - game_grid.py which contains a GameGrid class for modeling the game grid
  - Tetris_2048.py as the main program which enables the user move tetrominoes left, right or down on the game grid and includes a simple menu with an image and a start game button
  - images/menu_image.png which is the image (with a transparent background) shown in the simple game menu
- Example screenshots of the running program are given on the next slide.


stddraw window (r-click to save)
2048
Click Here to Start the Game

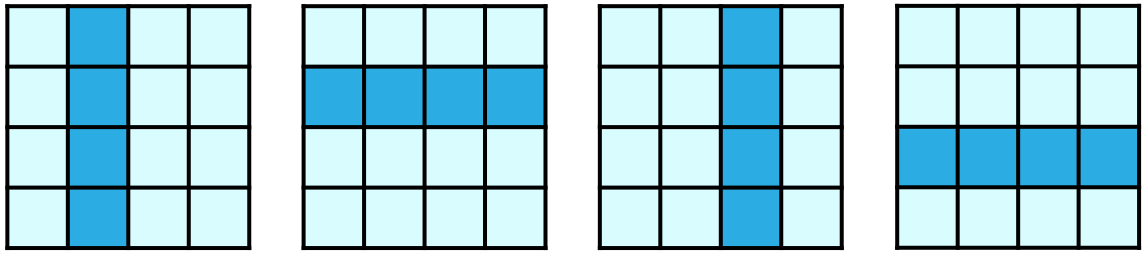



stddraw window (r-click to save)
Left Key (←)
Right Key (→)
Down Key (↓)

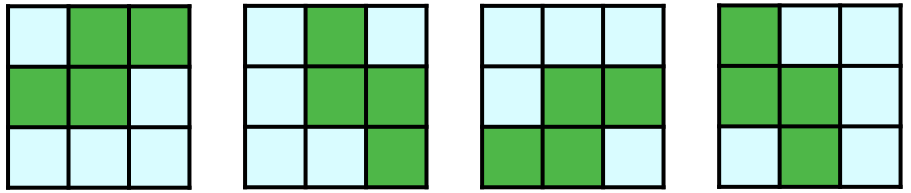
7 Different Types (Shapes) of Tetrominoes

- In the base code, a tile matrix is used to store the tiles on a tetromino based on its shape.
- This tile matrix is a square matrix (the number of the rows is equal to the number of the columns).
  - This makes it easier to implement a method for rotating tetrominoes.
- The first shapes given below for tetrominoes I and Z, and the shape given below for tetromino O are used in the base code for creating the tile matrices. You can also use the shapes below for implementing a simple method for rotating tetrominoes.

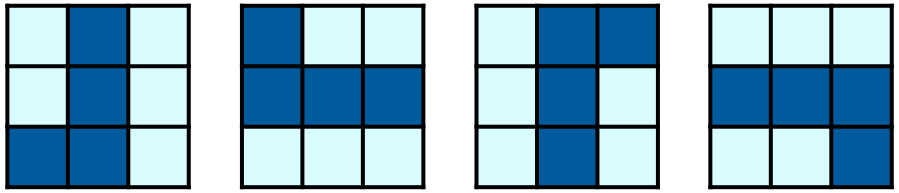
I



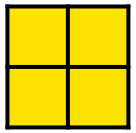
S



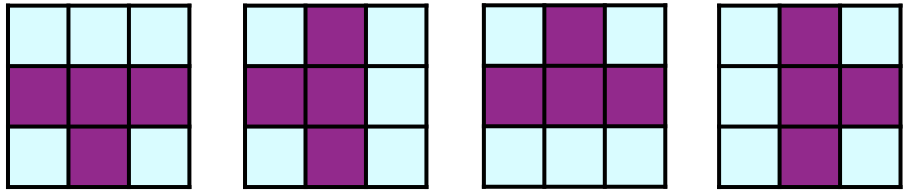
J



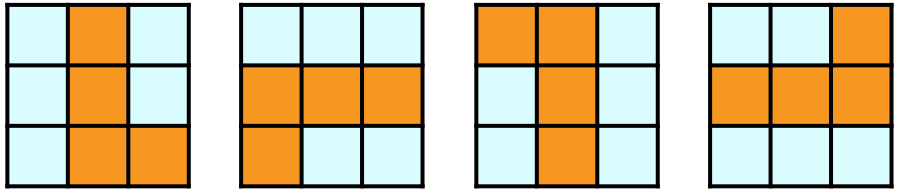
O



T

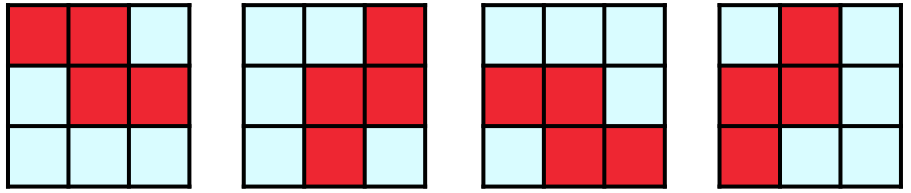


L



Note: Rotating tetromino O does not make a difference in Tetris, but it changes the positions of the numbered tiles in Tetris 2048.

Z



How to Use the stddraw Module

The stddraw module contains functions for creating a visual output and enabling user interactions by using the keyboard and the mouse.

How to Use the stddraw Module

- The stddraw (standard drawing) module contains functions for creating a visual output.
- It works with Python 3.4 or a newer version.
- It provides the following features for Python programs.
  - control functions for
    - setting the size of the drawing window in pixels
    - setting the range of the values for x and y coordinates on the drawing window
    - changing the thickness value and the color value used by the drawing functions
  - drawing points, lines and geometric shapes (circles, squares, rectangles and polygons)
  - displaying text
  - displaying images (pictures)
  - saving the current state of the window as an image file
  - creating animations
  - enabling user interactions by using the keyboard and the mouse

How to Use the stddraw Module

- **Control Functions:**

Function	Description
stddraw.setCanvasSize(width, height)	Set the size of the drawing window to width-by-height pixels (both width and height default to 512).
stddraw.setXscale(min, max)	Set the x-scale of the window such that the minimum x value is min, and the maximum x value is max. min defaults to 0 and max defaults to 1.
stddraw.setYscale(min, max)	Set the y-scale of the window such that the minimum y value is min, and the maximum y value is max. min defaults to 0 and max defaults to 1.
Note: The point defined by the min values set for the x-scale and the y-scale is at the bottom left corner of the window, and the point defined by the max values set for the x-scale and the y-scale is at the top right corner of the window.	
stddraw.setPenRadius(r)	Set the pen radius to r which affects the thickness of subsequent drawings (r defaults to 0.005). **Note:** If the pen radius is 0, then points and line widths will have the minimum possible size.
stddraw.setPenColor(c)	Set the pen color to c which affects the color of subsequent drawings (c defaults to color.BLACK). **Note:** c is an object of class Color in module color (color.py).

How to Use the stddraw Module

Basic Drawing Functions:

Function	Description
stddraw.point(x, y)	Draw a point at the location (x, y).
stddraw.line(x0, y0, x1, y1)	Draw a line from the location (x0, y0) to the location (x1, y1).

Shape Drawing Functions:

Function	Description
stddraw.circle(x, y, r)	Draw the outline (boundaries) of a circle with radius r centered at the location (x, y).
stddraw.square(x, y, r)	Draw the outline (boundaries) of a 2r-by-2r square centered at the location (x, y).
stddraw.rectangle(x, y, w, h)	Draw the outline (boundaries) of a rectangle with width w, height h and bottom left corner point (x, y).
stddraw.polygon(x, y)	Draw the outline (boundaries) of a polygon that connects the corner points (x[i], y[i]). x and y are lists that store the coordinates of the corner points.
Note: filledCircle(x, y, r), filledSquare(x, y, r), filledRectangle(x, y, w, h) and filledPolygon(x, y) draw filled shapes, not just outlines.	

How It Works:

- The stddraw module uses 2 drawing canvases: an invisible background canvas and a visible window canvas.
- All the drawings are made on the background canvas, and they become visible only by invoking the method stddraw.show(msec). This method copies all the drawings from the background canvas to the window canvas.

How to Use the stddraw Module

- **Functions for Drawing Text:**

Function	Description
stddraw.setFontFamily(f)	Set the font family to f (f defaults to 'Helvetica').
stddraw.setFontSize(s)	Set the font size to s (s defaults to 12).
stddraw.text(x, y, s)	Draw the given string s centered at (x, y).
stddraw.boldText(x, y, s)	Draw the given string s as a bold text centered at (x, y).

- **Function for Adding an Image (Picture) to the Drawing:**

Function	Description
stddraw.picture(pic, x, y)	Display the given picture pic centered at the location (x, y). x and y default to the center of the window. **Note:** pic is an object of class Picture in module picture (picture.py).

- **Function for Saving the Current Window as an Image File:**

Function	Description
stddraw.save(f)	Save the window canvas to a file named f.

How to Use the stddraw Module

- **Animation Functions:**
  - The stddraw module uses 2 drawing canvases: an invisible background canvas and a visible window canvas.
  - All the drawings are made on the background canvas, and they become visible only by invoking the method stddraw.show(msec). This method copies all the drawings from the background canvas to the window canvas.
  - To create the illusion of movement (animation), the three steps given below are repeated.
    - Clear the background canvas.
    - Draw something (e.g., some moving balls at their current positions).
    - Show the drawing and wait for a short while.

Function	Description
stddraw.clear(c)	Clear the background canvas to color c (c defaults to color.WHITE). **Note:** c is an object of class Color in module color (color.py).
stddraw.show(msec)	Display the drawing by copying the background canvas to the window canvas, and then wait for msec milliseconds. **Note:** msec defaults to infinity. In this case, the program waits until the window is closed by the user.

How to Use the stddraw Module

- **Functions for User Interaction via Keyboard and Mouse:**

Function	Description
stddraw.hasNextKeyTyped()	Return True if the queue of the keys the user typed is not empty, and False otherwise.
stddraw.nextKeyTyped()	Remove the first key from the queue of the keys that the user typed and return that key.
stddraw.clearKeysTyped()	Clear all the keys in the queue of the keys that the user typed.
stddraw.mousePressed()	Return True if the mouse has been left-clicked since the last time mousePressed() was called, and False otherwise.
stddraw.mouseX()	Return the x coordinate of the location at which the mouse has most recently been left-clicked.
stddraw.mouseY()	Return the y coordinate of the location at which the mouse has most recently been left-clicked.