Intro to Computer Science
CS-UH 1001, Spring 2022

Lecture 15 – Graphical User Interface (GUI) using Processing

Card, Deck and Hand class (ex_14.1.py)

1. Create a card class in OOP

   A card has the following attributes and methods:

   - A suit (Spades, Hearts, Diamonds, Clubs)
   - A rank (Ace, 2, 3, 4, …., 10, Jack, Queen, King)
   - A value (11, 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10)
   - A print method

2. Create a deck class

   A deck is a collection of all possible combinations of cards (52 cards). A deck has one attribute to hold the 52 cards and the following methods:

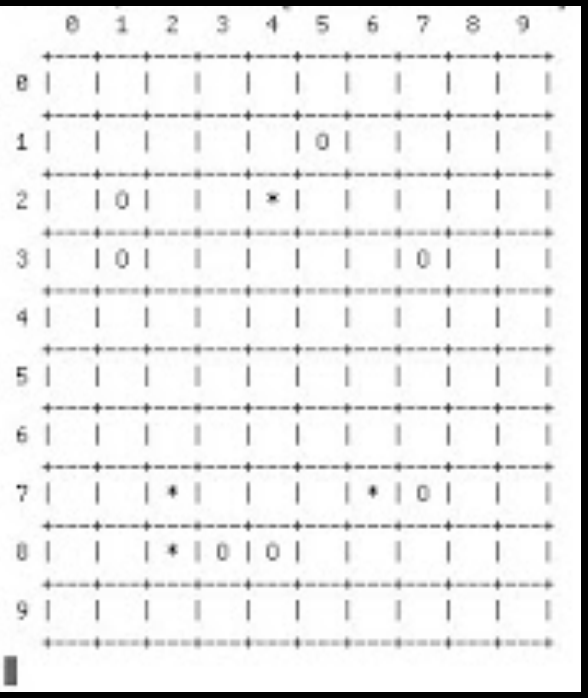
   - Printing the deck
   - Shuffling the deck (use random.shuffle(list))
   - Dealing a card

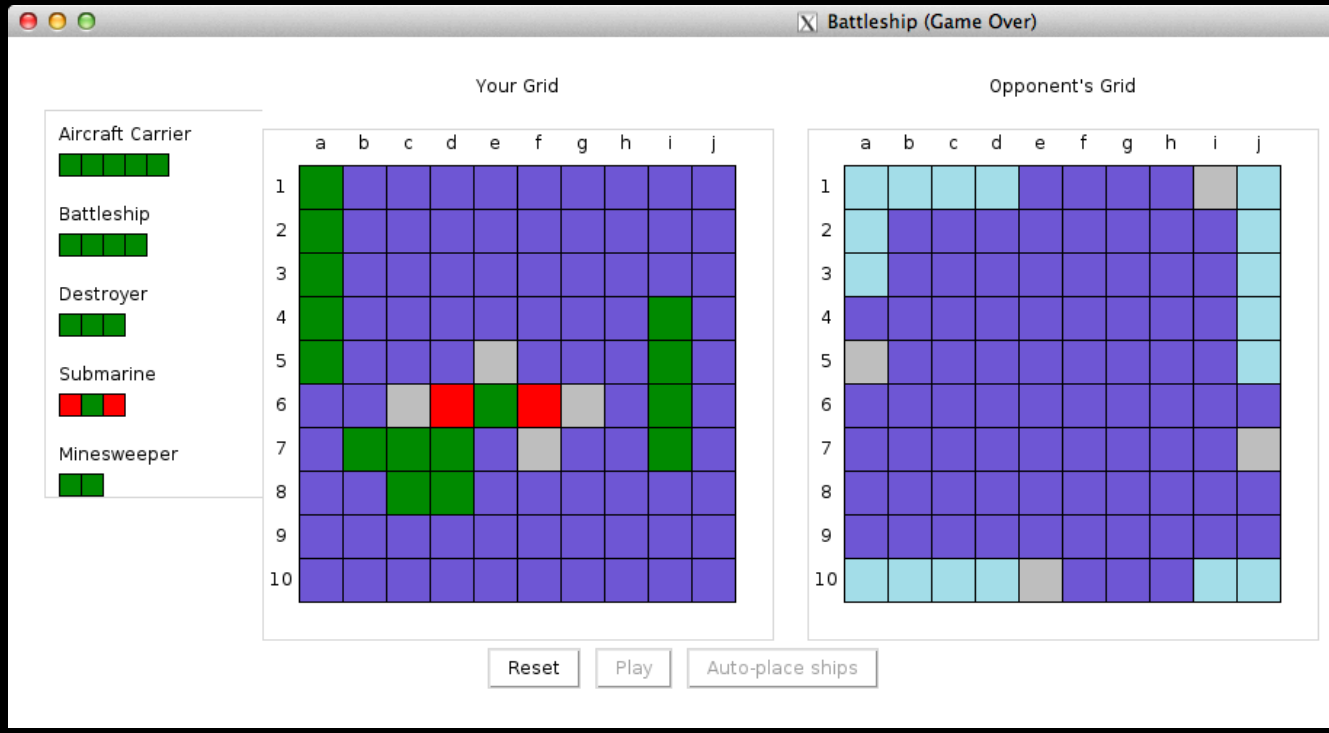
Create an empty Hand class that inherits from the Deck class, create 2 players (hands) and deal 2 cards for each from the Deck class

Graphical User Interface (GUI)

- A GUI allows the user to interact with a program/electronic device though icons and other visual indicators rather than text via command line
  - Your OS uses GUIs (Windows, MacOS, Linux)
  - Your programs use GUIs (MS Office, Calculator, etc.)

Graphical User Interface (GUI)



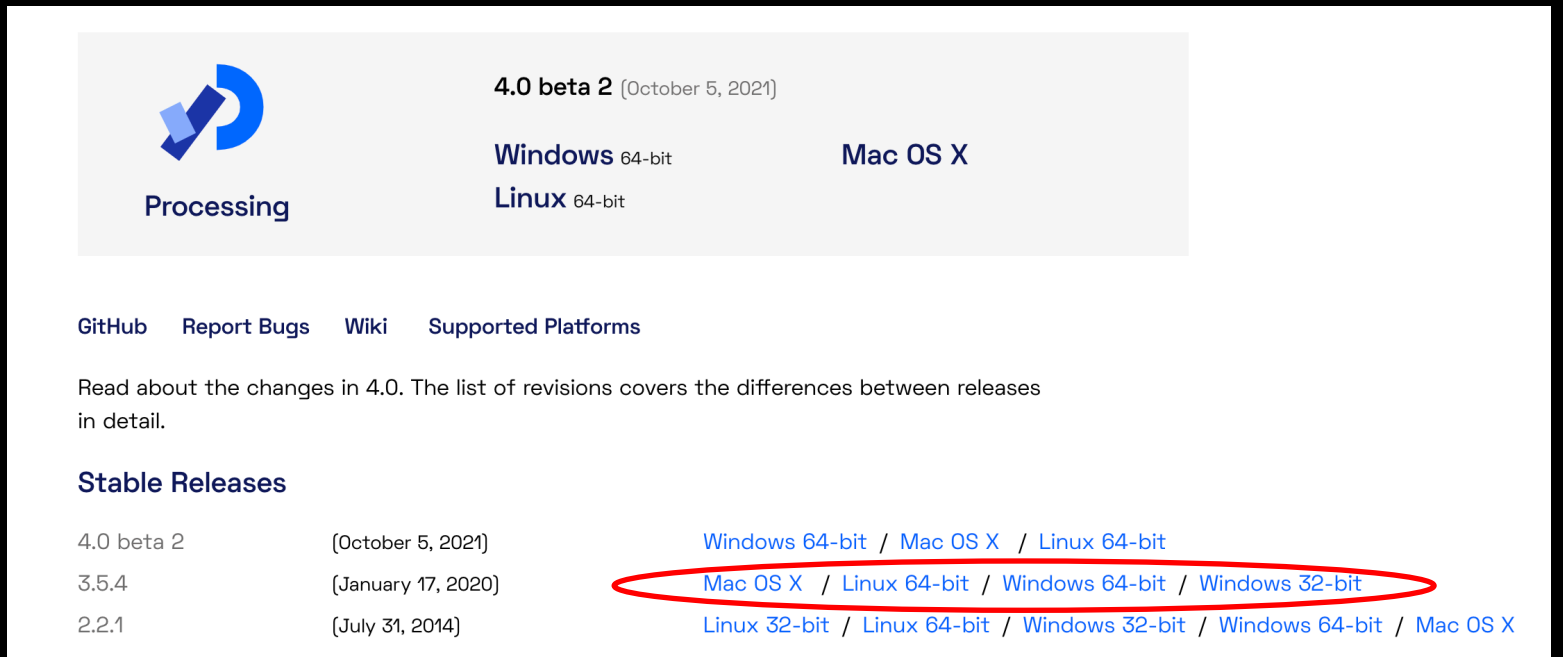


Processing 3.0

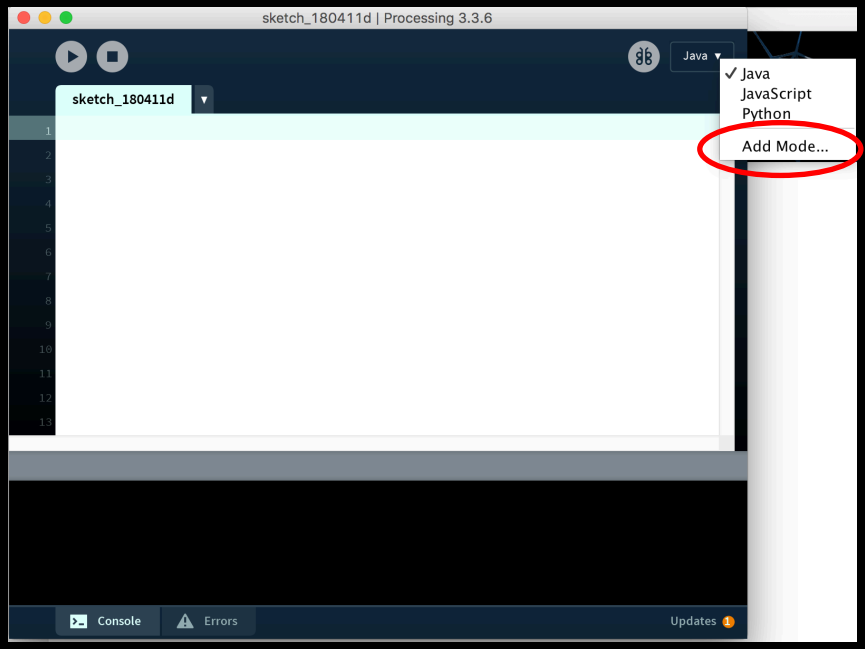
- Processing is an open source programming language and integrated development environment (IDE)
- It was built for electronic arts, new media art and visual design
- The project started in 2001 by Casey Reas and Benjamin Fry (formerly of the Aesthetics and Computation group at MIT media Lab)

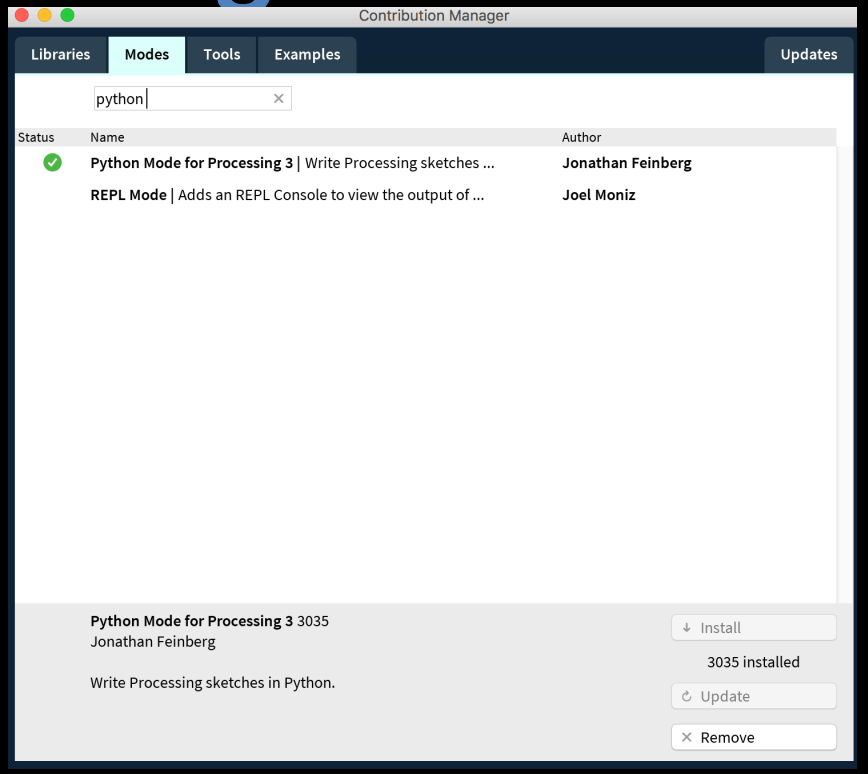
Installing Processing

- Please install Processing from: processing.org/download/
- Make sure you install version 3.5.4!!!



Installing Python Mode for Processing





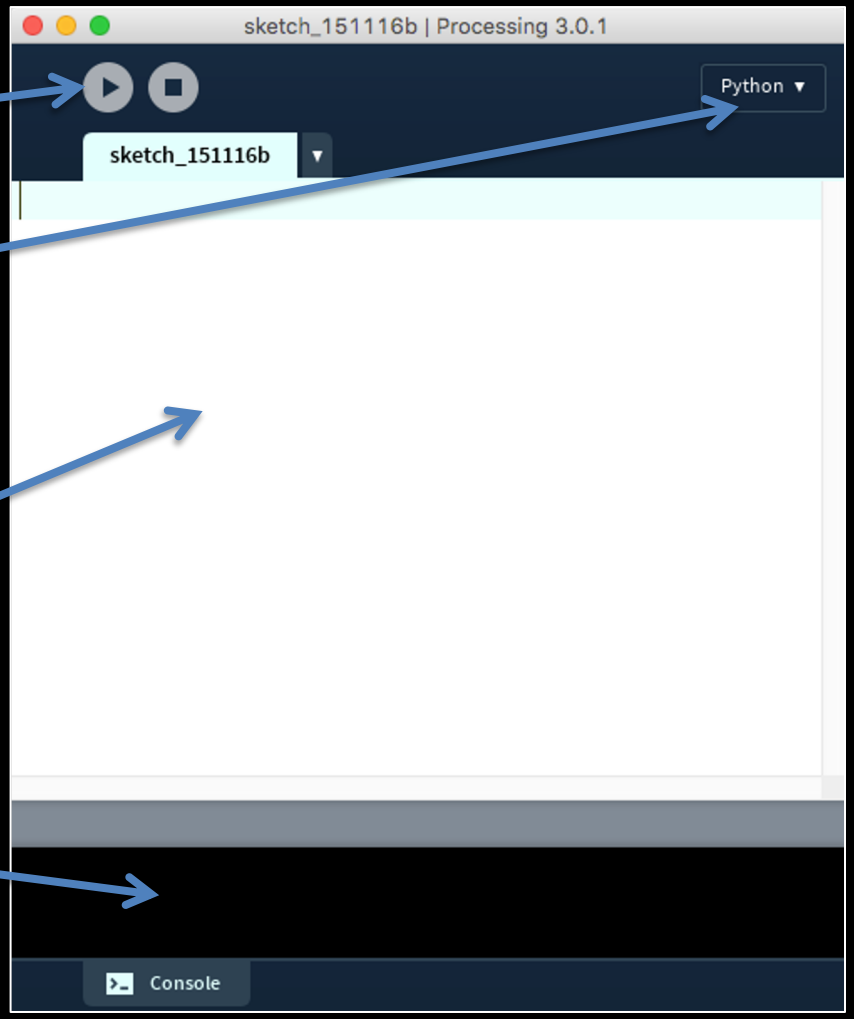
Install Python Mode

When installation finished, restart processing

Switch the mode to python

Processing IDE

- Run/Stop buttons
- Module
- Text editor
- Terminal output



Processing basic functions

- For every Processing code, these two functions must be in every Processing program:
  - setup()
  - draw()

setup() Function

- setup() the first functions that is executed when the program starts
- Used to define the initial environment properties such as window size, background color, etc.
  - Window screen size:
    - size(width, height): both width and height are in pixels
  - Window background color:
    - background(R,G,B): R,G, and B are the RGB color values, or 1 value for grayscale

Important: any variable declared within the setup() function will not be visible anywhere else (local function variable!)

draw() Function

- This function is called directly after the setup() function
- This function will continuously loop and executes the lines of code inside it
- The number of times the draw() function is executed per second can be defined by the frameRate() function
  - Default processing frame rate is 60 frames per second
  - Standard quality television animation uses 25 frames per second

Let's create our first program

```
def setup():
    size(640, 320)
    stroke(255, 255, 255)
    background(0, 0, 0)

def draw():
    line(0, 0, 100, 50)
```

Sets the color of the of the lines and borders around shapes

Draws a line between two points, takes 4 arguments (x1, y1, x2, y2)

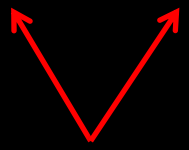
Mouse Functions

- In Processing there are a number of useful functions that are related to the mouse input:
  - mouseX: a variable that tells the current x-coordinate of the mouse position on the screen
  - mouseY: a variable that tells the current y-coordinate of the mouse position on the screen
  - mouseClicked(): a function definition that is called after a mouse button has been pressed and then released

Modify our first program

```
def setup():
    size(640, 320)
    stroke(255, 255, 255)
    background(0, 0, 0)

def draw():
    line(150, 25, mouseX, mouseY)
```



Gets the x and y location of the mouse

Basic Shapes

- Shapes:
  - line(x1, y1, x2, y2): draws a line from x1,y1 to x2,y2
  - rect(x, y, width, height): draws a rectangle
  - triangle(x1, y1 ,x2, y2, x3, y3): draws a triangle
  - ellipse(x, y, width, height): draws an ellipse

- Shape color:
  - fill(R, G, B): sets the color of the shape
  - noFill(): transparent background
  - stroke(R, G, B): sets the color of lines and border lines
  - strokeWeight(width): sets the width of the border lines

Note: the fill() and stroke() function must be called before creating the shape

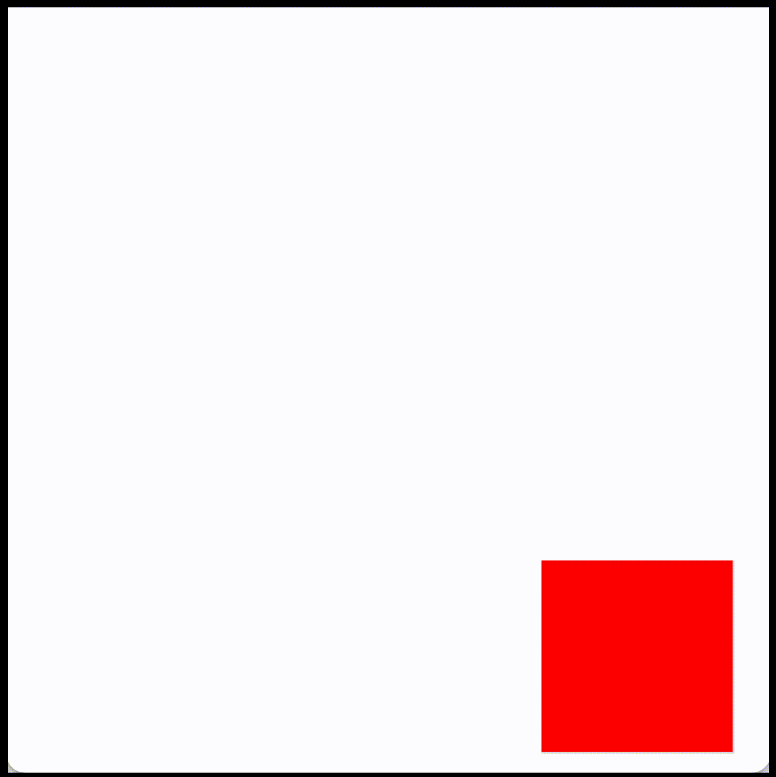
Example: Drawing an Ellipse

```
def setup():
    size(640, 320)
    stroke(255, 255, 255)
    background(0, 0, 0)

def draw():
    ellipse(150, 25, mouseX, mouseY)
```

Example: Using the Rectangle Class

- Use the rectangle class from last class to draw a rectangle that follows the mouse



Keyboard Functions

- In Processing there are a number of useful functions that are related to the keyboard input:
  - keyPressed(): a function that is called after a key has been pressed
    - key: a variable that holds the value of the most recent key press (a, b, c, etc.)
    - keyCode: a variable that holds the value of the most recent special key press (LEFT, RIGHT, DOWN, UP, etc.)
  - keyReleased(): a function that is called after a key has been released

Image Handling

- img = loadImage(path to the image on disk): load an image. It supports four image types: .png, .jpg, .gif, and .tga.
  - Use loadImage() only once to load the image into memory!

- image(img, x, y, width, height, x1, y1, x2, y2): used to display the loaded image on the screen
  - img: is the name of the variable used to load the image
  - x and y: the location of the upper left corner of the image on the screen
  - width and height (optional): resize/shrink/expand the image when displayed on the screen
  - x1,y1,x2, and y2 (optional): crop the loaded image
    - x1,y1 represent the upper left corner
    - x2,y2 represent the lower right corner

Image Loading Example

```
img = loadImage('nyu.jpg')

def setup():
    size(400, 400)
    background(0, 0, 0)

def draw():
    # syntax: image(img, x, y, width, height, x1, y1, x2, y2):
    image(img, 0, 0)
    image(img, 200, 0, 200, 200)
    image(img, 0, 250, 50, 50)
    image(img, 100, 250, 50, 50, 100, 100, 200, 200)
    image(img, 200, 250, 50, 50, 200, 200, 0, 0)
    image(img, 300, 250, 50, 50, 200, 0, 0, 200)
```

Other Useful Processing Functions

- There are lots of other useful functions in Processing

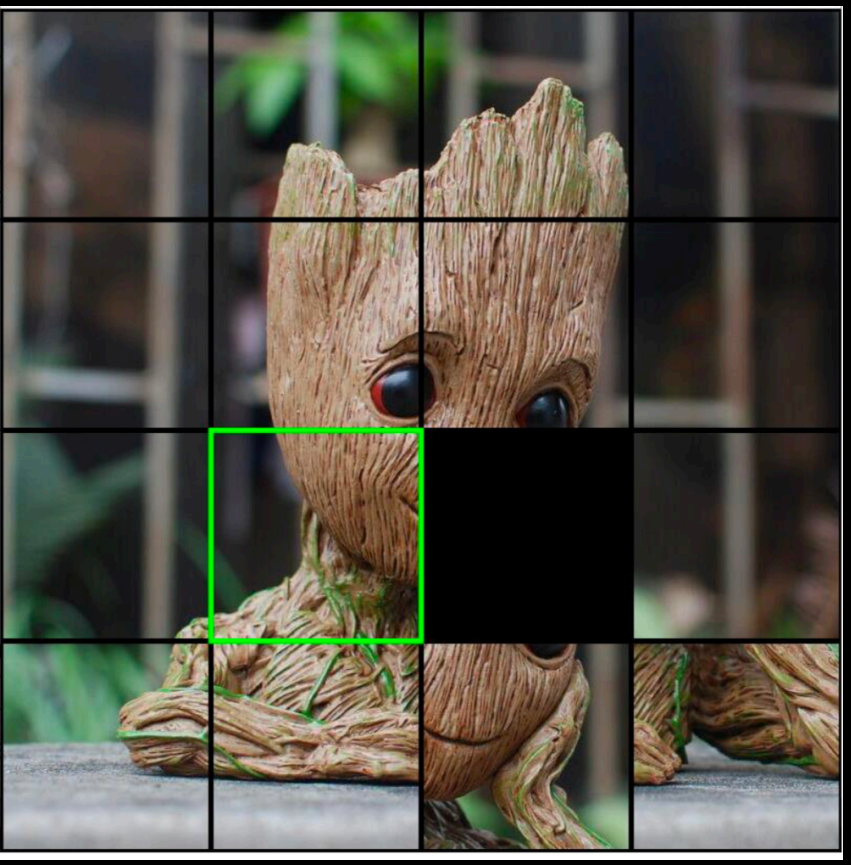
- Go to the link below for a full list and description of each:
  https://processing.org/reference/

Notice: all of the code described on the documentation is for Java (not Python).

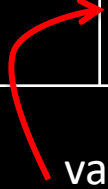
You need to know how to change the syntax to Python:

- Python has no { }, it uses indentation instead
- When declaring variables, Python doesn't specify which data type it will be
- There is no ; at the end of each line in Python

Lab: Sliding Puzzle



Tile attributes: row = 0 col = 0 value = 0 image = 0.png	Tile attributes: row = 0 col = 1 value = 1 image = 1.png	…	Tile attributes: row = 0 col = 3 value = 3 image = 3.png
Tile attributes: row = 1 col = 0 value = 4 image = 4.png	…	…	…
…	…	…	…
…	…	…	Tile attributes: row = 3 col = 3 value = 15 image = 15.png



value = row * NUM_COLS + col