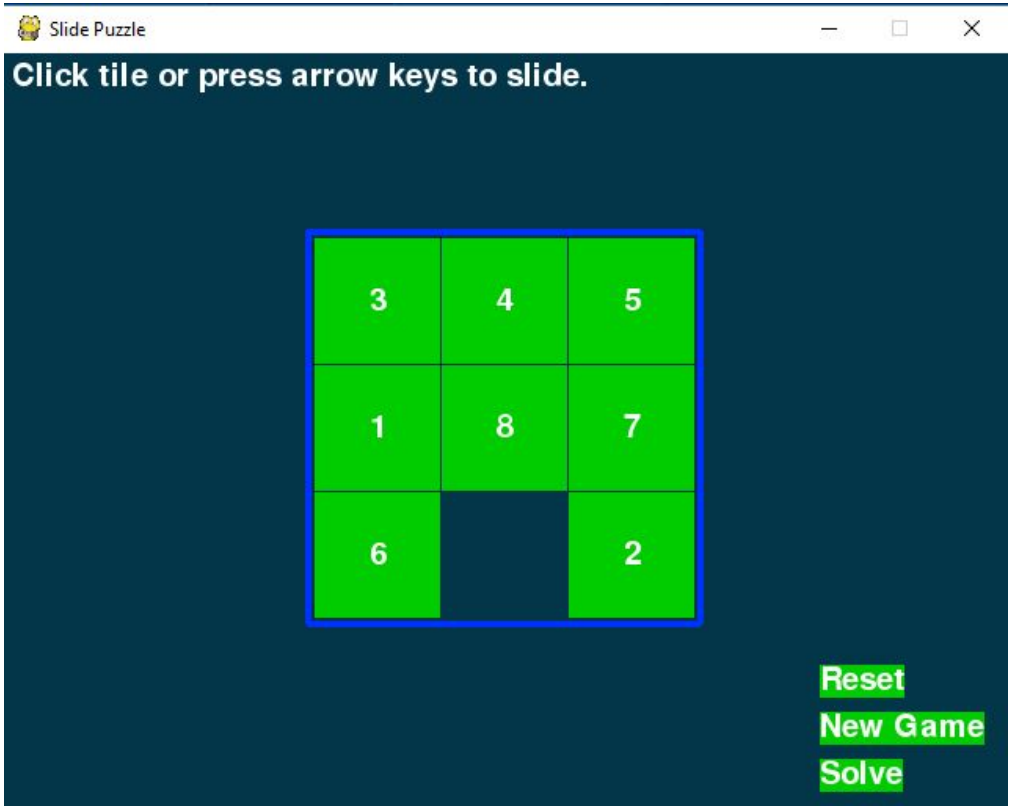
Sliding Puzzle

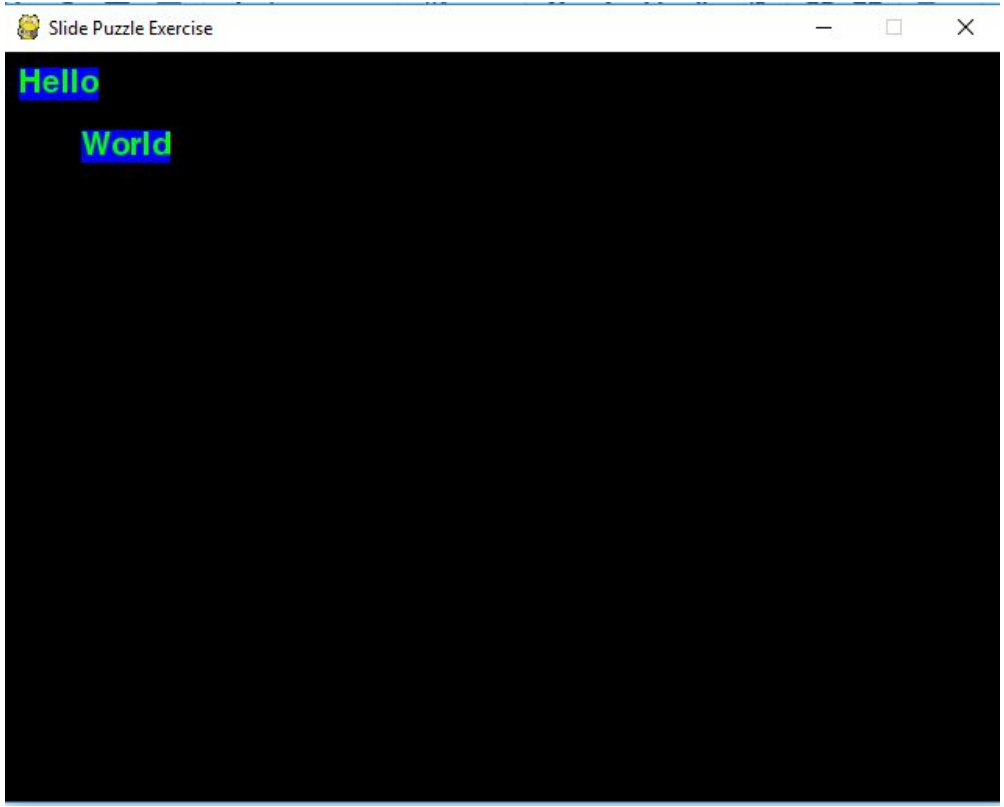
Making Game with Python
3/31/2019


Slide Puzzle
Click tile or press arrow keys to slide.
3
4
5
1
8
7
6
2
Reset
New Game
Solve

Features of the Game

- Quit: ‘X’ sign, ESC Key
- Words: Reset, New Game, Solve, Message
- Board:
  - Mouse
  - Arrow Keys: left, right, up, down
  - Letter Keys: A W S D

Project 1: Word Game



Main function (1)

```
def main(FPS=10):
    global BASICFONT

    pygame.init()
    FPSCLOCK = pygame.time.Clock()
    DISPLAYSURF = pygame.display.set_mode((640, 480))
    pygame.display.set_caption('Slide Puzzle Exercise')
    BASICFONT = pygame.font.Font('freesansbold.ttf', 20)

    textColor = (0, 255, 0)
    textBGColor = (0, 0, 255)
    helloSurf, helloRect = makeText('Hello', textColor, textBGColor, 10, 10)
    worldSurf, worldRect = makeText('World', textColor, textBGColor, 50, 50)
```

Main function (2)

```
while True:
    DISPLAYSURF.fill((0,  0, 0))
    DISPLAYSURF.blit(helloSurf, helloRect)
    DISPLAYSURF.blit(worldSurf, worldRect)
    for event in pygame.event.get(): # event handling loop
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
        elif event.type == MOUSEBUTTONUP:
            # check if the user clicked on an option button
            if helloRect.collidepoint(event.pos):
                textSurf, textRect = makeText('Hello is clicked', textColor, textBGColor, 100, 10)
                DISPLAYSURF.blit(textSurf, textRect)
            elif worldRect.collidepoint(event.pos):
                textSurf, textRect = makeText('World is clicked', textColor, textBGColor, 150, 50)
                DISPLAYSURF.blit(textSurf, textRect)
    pygame.display.update()
    FPSCLOCK.tick(FPS)
```

makeText function

```
def makeText(text, color, bgcolor, top, left):
    # create the Surface and Rect objects for some text.
    textSurf = BASICFONT.render(text, True, color, bgcolor)
    textRect = textSurf.get_rect()
    textRect.topleft = (top, left)
    return (textSurf, textRect)
```

Entry Point

```
if __name__ == '__main__':
    if len(sys.argv) > 1:
        main(int(sys.argv[1]))
    else:
        main()
```

Windows: py slidepuzzle_exercise.py 10

Mac: python3 slidepuzzle_exercise.py 10

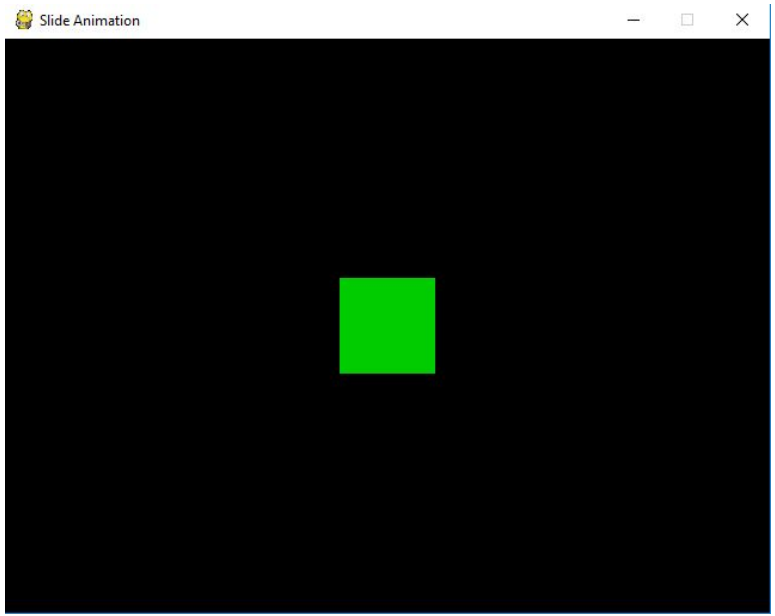
Python 3.7 IDE

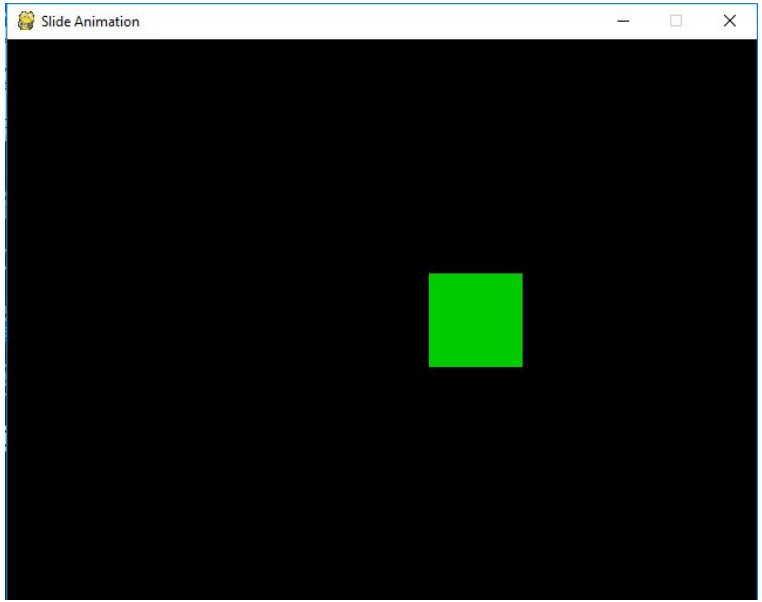
Import os

os.chdir(your_working_directory)

os.popen(‘py slidepuzzle_exercise.py 10’).read()

Project 2: Slide Animation





```
TILESIZE = 80
WINDOWWIDTH = 640
WINDOWHEIGHT = 480
FPS = 30

BGCOLOR =   (  0,   0,   0)
TILECOLOR = (  0, 204,   0)
NUM_ANAMATION = 8
SPEED = TILESIZE/NUM_ANAMATION

def main():
    global FPSCLOCK, DISPLAYSURF
    pygame.init()
    FPSCLOCK = pygame.time.Clock()
    DISPLAYSURF = pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT))
    pygame.display.set_caption('Slide Animation')

    left = WINDOWWIDTH/2 - TILESIZE/2
    top = WINDOWHEIGHT/2 - TILESIZE/2
```

```
while True: # main game loop
    DISPLAYSURF.fill(BGCOLOR)
    pygame.draw.rect(DISPLAYSURF, TILECOLOR, (left, top, TILESIZE, TILESIZE))
    for event in pygame.event.get(): # get all the QUIT events
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
        elif event.type == KEYUP:
            # check if the user pressed a key to slide a tile
            if event.key == K_LEFT:
                left, top = slideAnimation(left, top, -SPEED, 0)
            elif event.key == K_RIGHT:
                left, top = slideAnimation(left, top, SPEED, 0)
            elif event.key == K_UP:
                left, top = slideAnimation(left, top, 0, -SPEED)
            elif event.key == K_DOWN:
                left, top = slideAnimation(left, top, 0, SPEED)

    pygame.display.update()
    FPSCLOCK.tick(FPS)
```

```
def isValidMove(left, top):
    return left >=0 and left+TILESIZE <= WINDOWWIDTH \
            and top >= 0 and top+TILESIZE <= WINDOWHEIGHT

def slideAnimation(left, top, speedx, speedy):
    for i in range(NUM_ANAMATION):
        nextx = left + speedx
        nexty = top + speedy
        # animate the tile sliding over
        if isValidMove(nextx, nexty):
            DISPLAYSURF.fill(BGCOLOR)
            left, top = nextx, nexty
            pygame.draw.rect(DISPLAYSURF, TILECOLOR, (left, top, TILESIZE, TILESIZE))
            pygame.display.update()
            FPSCLOCK.tick(FPS)
        else:
            break
    return left, top

if __name__ == '__main__':
    main()
```

Exercise:

```
# generate 2-D array
def generate_number_board(n):
    board = []
    counter = 1
    for i in range(n):
        row = []
        for j in range(n):
            row.append(counter)
            counter += 1
        board.append(row)

    print(board)
```