CHEATSHEET

Sprite

```
# A Sprite is any in-game object with an animation, position, and size
# Sprites will automatically be drawn every loop in the update() function

sprite = Sprite(Animation, Position, size)

# Once you have a Sprite variable, it can be modified or read using get or set functions

sprite.set_anim(different_animation) # set a new animation for the heart sprite
sprite.set_pos(x, y) # set the center position of heart to (x, y)
sprite.get_pos() # returns the current center position of the sprite -> Position
sprite.clicked() # check if sprite is clicked
sprite.hovered() # check if the sprite is being hovered over
sprite.set_size(100, 100) # scale the heart to 100px by 100px
sprite.get_width() -> returns the width of heart
sprite.get_height() -> returns the height of heart
sprite.kill() # remove the sprite
sprite.move(Direction)
sprite.stop() # stop the sprite
```

Position

Position.x
Position.y

Animation

```
# an Animation is a representation of a spritesheet exported from Piskel or other software

animation = Animation("spritesheet.png", fps, columns, rows, num_of_frames)
```

General Examples

```
# initialize will create the window for our game!
initialize(screen_width, screen_height, title)

# update should be called each loop
update()

# set background color to blue
set_background_color( (0, 0, 255) )

get_mouse_pos() # get the coordinate position of the mouse
get_mouse_x()
get_mouse_y()

mouse_clicked() # return True if the mouse is being clicked
get_keys() # get list of the keys being pressed
quit() # quit and close the game window
wait(s) # wait for s seconds, then return True

check_collision(a, b) # return True if sprite a collides with b
```

Colors

BLUE, GREEN, RED,
WHITE, BLACK

Create your own color with RGB:

CUSTOM = (230, 100, 35)

Concepts

Game Loop

Our game loop is a **`while True:`** loop which runs 60 times each second. At the bottom of the loop, the update function draws all sprites.

Position

Position represents something's (x, y) coordinates within the window, for ease of placement, Sprite.set_pos() uses the center of the sprite.

Collisions

A collision represents the overlap of a position with an sprite's rectangular area or a sprite's area with another sprites area.