Objectives:

* Use sprite sheets for animating sprites, multiple rows and single row
* Use Pygame libraries to clip sprite sheet into sequential, individual frames
* Use command line arguments in a batch file to steer the program to use different resources
* Use the os package in python to better organize the code

*Goal:*

*Search and find an animated sprite which appeals to you. Or use one of the supplied sprite sheets. Then find a background image over which you will place your animated sprite and show the animation.*

In Project 1 we will use the PyGame Animation class which creates an animation from a list of images. In chapter 12 the explosion was provided by the individual frames in the resources folder. Most developers, however, don’t provide animated characters in this format. They develop their game resources in a form called sprite sheets (SnailBait). We won’t be using such a complex sprite sheet, though. We can find simple single-figure sprite sheets (Stick Man Walking). In order to use this simpler sprite sheet, we need to create a list of the individual frames, from this image.



Stick Man Walking



**spritesheet from SnailBait**

The dimensions of the Walking image are 1200 pixels wide by 148 pixels high. A little calculation tells us that the eight individual frames should be 150 pixels by 148 pixels.

Rather than use the more tedious process of using an image processor to cut each image and then read them into a list in our Python script, we should use our Python tools to accomplish this automatically. Pygame has some nice utilities for the ‘image’ class, primarily the subsurface method. Subsurface cuts a portion of the image by placing the pixel address at the upper left of the intended frame, ‘w’ pixels wide and ‘h’ pixels high. A for-in loop works nicely to march through the image, cutting the individual frames. The frames are then added to the list ‘images’. Now we have a list of images that can be added to the Animation class of livewires. The Python script above uses the ‘os’ library to search for an image file included in the *sprite* folder in our folder hierarchy.

import pygame, os, livewires

def load\_sliced\_sprites(w, h, filename):

images = []

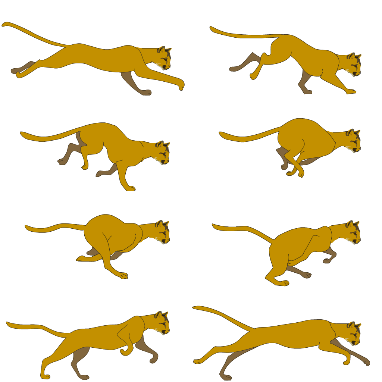
master\_image = pygame.image.load(os.path.join('sprites', filename)).convert\_alpha()

master\_width, master\_height = master\_image.get\_size()

for i in range(int(master\_width/w)):

images.append(master\_image.subsurface((i\*w,0,w,h)))

return images

Not all sprite sheets have the animation laid out in a single row, as in the sheet *Running Cat*. The algorithm can be improved by adding an outer loop to advance the cutting down each row.

master\_width, master\_height = master\_image.get\_size()

for j in range(int(master\_height/h)):

for i in range(int(master\_width/w)):

images.append(master\_image.subsurface((i\*w,j\*h,w,h)))

return images

For Project 1, you are asked to place the animated sprite, of your choice, on a background, of your choice. Make the animation run for several cycles.

[Challenge: use an image processor like Maya or Gimp to create an alpha (transparency) channel.]

