pyGame Lecture #1
(drawing.py and drawstuff.py)

INTRODUCTION TO PROGRAMMING USING PYGAME

STEP 1: Importing Modules and Initialization

All the Pygame functions that are required to implement features like graphics and sound are stored in the Pygame module and sys module. These modules must be imported into our namespace for us to access the functions inside these modules.

To do this, we use the statement:

```
import pygame, sys
```

Note: When we import a module, we gain access to the sub-modules too.

Note: There are two formats to import modules:

1. import modulename
2. from modulename import *

Normally, when we use import modulename, we can refer to the functions inside by calling modulename.functionname.

However, using from modulename import *, we can refer to the function simply by its name without the modulename prefix.

The module pygame.locals contains many constant variables. So, to reduce the typing workload, we import pygame.locals in the second format.

```
from pygame.locals import *
```

```
pygame.init()
```

It is a function call, that is performed before any other Pygame function call. It needs to be called first in order for many other Pygame functions to work.

STEP 2: Setting the Window Size

```
DISPLAYSURF = pygame.display.set_mode((700, 500))
```

The set_mode() function accepts a tuple of two integers as its argument and returns a pygame.Surface object for the display window. We pass a tuple value of two integers to the function: (700, 500). This tuple tells the set_mode() function how wide and how high to make the window in pixels. (700, 500) will make a window with a width of 700 pixels and height of 500 pixels.

Note: Make sure to pass a sequence, i.e., a tuple of two integer values to the set_mode() function, not two integers.

STEP 3: Adding a Window Caption

pygame.display.set_caption('Summer Camp on fleek')

This sets the caption text that will appear at the top of the window by calling the pygame.display.set_caption() function. The string value 'Summer Camp on fleek' is passed in this function call to make that text appear as the caption.

STEP 4: Game Loops and Events

while True: # main game loop

for event in pygame.event.get():

This is a while loop that has a condition of simply the value True. This means that it never exits and the only way the program execution will ever exit the loop is if a break statement is executed or sys.exit() (which terminates the program). If a loop like this was inside a function, a return statement will also move execution out of the loop (as well as the function too).

A **game loop** (also called a **main loop**) is a loop where the code does three things:

1. Handles events.
2. Updates the game state.
3. Draws the game state to the screen.

A game state is just a way of referring to a set of values for all the variables in a game program. Since a game is affected by passage of time or outside events like mouse click or a key press, the program needs to be constantly checked for events and these evets are checked for using the pygame.event.get() function.

STEP 5: Program Termination

The QUIT Event and pygame.quit() Function

```
if event.type == QUIT:
    pygame.quit()
sys.exit()
```

This statement checks if the event type is QUIT. If yes, then it calls the pygame.quit() function and sys.exit() to terminate the program.

Some Useful Pygame Functions

1. pygame.draw

Pygame module to draw shapes: (pygame.draw)

Function name	Functionality
pygame.draw.line	Draws a straight line segment
pygame.draw.circle	Draws a circle around a point
pygame.draw.rect	Draws a rectangle shape
pygame.draw.ellipse	Draws a round shape inside a rectangle
pygame.draw.polygon	Draws a shape with any number of sides
pygame.draw.arc	Draws a partial section of an ellipse
pygame.draw.lines	Draws multiple line segments

2. pygame.Color

Define the colors used in RGB format as follows:

COLOR	R	G	B
Red	255	0	0
Green	0	255	0
Blue	0	0	255
Black	0	0	0
White	255	255	255

3. pygame.time

Function name	Functionality
clock	To track the time used by a frame
delay	Pauses for a given time (returns the number of milliseconds)
get_ticks	Time since pygame.time was imported (in ms)
set_timer	Sets the timer for an event (the event then gets placed on an event queue)
wait	Pauses for a given time (like delay but less accurate)