Loops

Loops

Loops are used to repeat code as long as a condition is met

For Loops

For loops run a set number of times

```
for x in range(3):
    print "Hi"
```

Output

Hi
Hi
Hi

For Loops

The y is a variable that represents 0 through num-1

```
for y in range(num):
```

It repeats loop until y is no longer less than num

```
y = 0
loop if y < num:
    code
    y += 1
```

For Loops

```
for z in range(5):
    print z
```

Output

```
0
1
2
3
4
```

For Loops

If range is given two numbers – range(a, b) – r will start at a and end at b – 1

```
for r in range(a, b):
```

It repeats loop until r is no longer less than b

```
r = a
loop if r < b:
    code
    r += 1
```

For Loops

```
for r in range(12, 16):
    print r
```

Output

```
12
13
14
15
```

For Each Loops

For Each loops run through every character in a string

```
for c in "Hello":
    print c
```

Output

```
H
e
l
l
o
```

Break keyword

The keyword break allows you to stop a loop early.

```
word = "Computer Science"
for w in word:
    if w == "u":
        break
    print w
```

Output

```
C
o
m
p
```

open fors.py

While Loops

While loops repeat code as long as a condition is true

```
answer = "yes"

while answer == "yes":
    answer = raw_input("Keep Looping?")
```

While Loops

```
x = 0

while x < 5:
    print x
    x += 1
```

Output

```
0
1
2
3
4
```

While Loops

The pygame runners use an infinite while loop to constantly update the screen and check for key presses

```
while True:
    for event in pygame.event.get():
        if event.type==QUIT:
            pygame.quit()
            sys.exit()
```

open while.py

Nested Loops

Loops can go inside other loops.

```
for x in range(3):
    print "x is", x
    for y in range(2):
        print "y is", y
    print
```

Nested Loops

```
for x in range(3):
    print "x is", x
    for y in range(2):
        print "y is", y
    print
```

Output

```
x is 0
y is 0
y is 1

x is 1
y is 0
y is 1

x is 2
y is 0
y is 1
```

Nested Loops

```
for x in range(3):
    for y in range(3):
        print "(", x, ",", y , ")",
    print
```

Output

```
( 0 , 0 )( 0 , 1 )( 0 , 2 )
( 1 , 0 )( 1 , 1 )( 1 , 2 )
( 2 , 0 )( 2 , 1 )( 2 , 2 )
```

open nested.py

Start work on Loops Labs

Pygame

frequently used methods

Name	Use
pygame.image.load(file)	This loads an image from a file and returns it as a surface
surface.blit(img, (x,y))	This draws an surface on another surface
pygame.draw.rect(Surface, color, Rect, width=0)	Draws a rectangle on a surface
pygame.draw.circle(Surface, color, pos, radius, width=0)	Draws a circle on a surface

Pygame methods

img = pygame.image.load(*"dude.gif"*)

pygame.image.load() loads an image as a surface

"dude.gif" is the file name of the image. It should be stored in the folder with your code

Pygame methods

window.blit(img, (x, y))

Draws the surface *img* onto the surface *window*.

***x* and *y* are the coordinates where the upper left corner of the image will be placed**

Pygame methods

pygame.draw.rect(window, (r, g, b),
(x, y, width, height))

Draws a rectangle onto surface *window*.

***r*, *g*, *b* represent a color as (red, green, blue). Each one can be a number from 0 - 255**

Pygame methods

pygame.draw.rect(window, (r, g, b),
(x, y, width, height))

***x* and *y* are the coordinates where the upper left corner of the rectangle will be placed**

***Width* and *height* are the width and height of the rectangle you want to draw**

Pygame methods

pygame.draw.circle(window, (r, g, b),
(x, y), rad)

This draws a circle at with it's upper left corner at (*x, y*) with a radius of rad

Random

frequently used methods

Name	Use
random.randint(a, b)	Get a random integer between a and b
random.random()	Get a random number between 0.0 and 1.0