Lista de Exercícios 2 (Slides: Python 2)

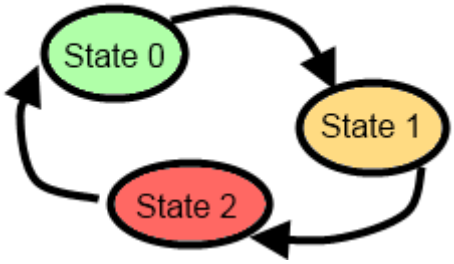
Instruções:

a) Os exercícios são individuais e devem ser entregues como projetos no GitHub.

b) Enviar o link do site no GitHub através do Google Class.

c) Data final para entrega: 5/05/2020

A simple state machine that we encounter often is a traffic light. Here is a state diagram which shows that the machine continually cycles through three different states, which we've numbered 0, 1 and 2.



We're going to build a program that uses a turtle to simulate the traffic lights. There are three lessons here. The first shows off some different ways to use our turtles. The second demonstrates how we would program a state machine in Python, by using a variable to keep track of the current state, and a number of different `if` statements to inspect the current state, and take the actions as we change to a different state. The third lesson is to use events from the keyboard to trigger the state changes.

```
import turtle           # Tess becomes a traffic light.

turtle.setup(400,500)
wn = turtle.Screen()
wn.title("Tess becomes a traffic light!")
```

```
wn.bgcolor("lightgreen")
tess = turtle.Turtle()


def draw_housing():
    """ Draw a nice housing to hold the traffic lights """
    tess.pensize(3)
    tess.color("black", "darkgrey")
    tess.begin_fill()
    tess.forward(80)
    tess.left(90)
    tess.forward(200)
    tess.circle(40, 180)
    tess.forward(200)
    tess.left(90)
    tess.end_fill()


draw_housing()

tess.penup()
# Position tess onto the place where the green light should be
tess.forward(40)
tess.left(90)
tess.forward(50)
# Turn tess into a big green circle
tess.shape("circle")
tess.shapesize(3)
tess.fillcolor("green")

# A traffic light is a kind of state machine with three states,
# Green, Orange, Red.  We number these states  0, 1, 2
# When the machine changes state, we change tess' position and
# her fillcolor.

# This variable holds the current state of the machine
state_num = 0


def advance_state_machine():
    global state_num
    if state_num == 0:       # Transition from state 0 to state 1
        tess.forward(70)
        tess.fillcolor("orange")
        state_num = 1
    elif state_num == 1:     # Transition from state 1 to state 2
        tess.forward(70)
        tess.fillcolor("red")
        state_num = 2
```

```
55      else:                    # Transition from state 2 to state 0
56          tess.back(140)
57          tess.fillcolor("green")
58          state_num = 0
59
60  # Bind the event handler to the space key.
61  wn.onkey(advance_state_machine, "space")
62
63  wn.listen()                      # Listen for events
64  wn.mainloop()
```

The new Python statement is at line 47. The `global` keyword tells Python not to create a new local variable for `state_num` (in spite of the fact that the function assigns to this variable at lines 51, 55, and 59). Instead, in this function, `state_num` always refers to the variable that was created at line 43.

What the code in `advance_state_machine` does is advance from whatever the current state is, to the next state. On the state change we move tess to her new position, change her color, and, of course, we assign to `state_num` the number of the new state we've just entered.

Each time the space bar is pressed, the event handler causes the traffic light machine to move to its new state.

1) Add some new key bindings to the first sample program:

   Pressing keys R, G or B should change tess' color to Red, Green or Blue.

   Pressing keys + or - should increase or decrease the width of tess' pen. Ensure that the pen size stays between 1 and 20 (inclusive).

Handle some other keys to change some attributes of tess, or attributes of the window, or to give her new behaviour that can be controlled from the keyboard.

2) Change the traffic light program so that changes occur automatically, driven by a timer.

3) Create a module named `wordtools.py` with our test scaffolding in place.

Now add functions to these tests pass:

```
test(cleanword("what?") == "what")
test(cleanword("'now!'") == "now")
test(cleanword("?+='w-o-r-d!,@$()'") ==  "word")

test(has_dashdash("distance--but"))
test(not has_dashdash("several"))
test(has_dashdash("spoke--"))
test(has_dashdash("distance--but"))
test(not has_dashdash("-yo-yo-"))

test(extract_words("Now is the time!  'Now', is the time?
Yes, now.") ==

['now','is','the','time','now','is','the','time','yes','n
ow'])
test(extract_words("she tried to curtsey as she spoke--
fancy") ==

['she','tried','to','curtsey','as','she','spoke','fancy']
)

test(wordcount("now",
["now","is","time","is","now","is","is"]) == 2)
test(wordcount("is",
["now","is","time","is","now","the","is"]) == 3)
test(wordcount("time",
["now","is","time","is","now","is","is"]) == 1)
test(wordcount("frog",
["now","is","time","is","now","is","is"]) == 0)

test(wordset(["now", "is", "time", "is", "now", "is",
"is"]) ==
      ["is", "now", "time"])
test(wordset(["I", "a", "a", "is", "a", "is", "I", "am"])
==
      ["I", "a", "am", "is"])
test(wordset(["or", "a", "am", "is", "are", "be", "but",
"am"]) ==
      ["a", "am", "are", "be", "but", "is", "or"])

test(longestword(["a", "apple", "pear", "grape"]) == 5)
test(longestword(["a", "am", "I", "be"]) == 2)
test(longestword(["this","supercalifragilisticexpialidoci
ous"]) == 34)
```

```
    test(longestword([ ]) == 0)
```

4) Modify the queens program (seção 14.8 do livro texto) to solve some boards of size 4, 12, and 16. What is the maximum size puzzle you can usually solve in under a minute?

5) Add a method `reflect_x` to `Point` which returns a new `Point`, one which is the reflection of the point about the x-axis. For example, `Point(3, 5).reflect_x()` is (3, -5)

6) Add a method `slope_from_origin` which returns the slope of the line joining the origin to the point. For example,

```
>>> Point(4, 10).slope_from_origin()
2.5
```

What cases will cause this method to fail?

7) The equation of a straight line is "y = ax + b", (or perhaps "y = mx + c"). The coefficients a and b completely describe the line. Write a method in the `Point` class so that if a point instance is given another point, it will compute the equation of the straight line joining the two points. It must return the two coefficients as a tuple of two values. For example,

```
>>> print(Point(4, 11).get_line_to(Point(6, 15)))
>>> (2, 3)
```

8) Crie um programa que ilustre o uso de métodos abstratos, métodos de classe e métodos estáticos. O programa deverá mostrar as vantagens de cada tipo de método.

9) Crie extensões no exemplo de classes de polígonos regulares de modo a avaliar os diferentes MROs possíveis;

10) Pesquise o assunto Decoradores. Desenvolva um Decorador exemplo.

11) Crie um exemplo de função com lista de argumentos e dicionário de argumentos.

12) Crie um exemplo que ilustre o conceito de polimorfismo com Python. O exemplo deverá também mostrar o que é Duck Typing.