Actividad 05 – Clases y objetos

Hernández Lomelí Diego Armando

Seminario de algoritmia

Lineamientos de actividad.

- [ ] El reporte está en formato Google Docs o PDF.
- [ ] El reporte sigue las pautas del Formato de Actividades .
- [ ] El reporte tiene desarrollada todas las pautas del Formato de Actividades.
- [ ] Se muestra la captura de pantalla de los datos antes de usar el método `agregar_inicio()` y la captura de pantalla del método `mostrar()` después de haber utilizado el método `agregar_inicio()`.
- [ ] Se muestra la captura de pantalla de los datos antes de usar el método `agregar_final()` y la captura de pantalla del método `mostrar()` después de haber utilizado el método `agregar_final()`.

Desarrollo

El inicio de esta actividad se da en el archivo **"Particula.py"**, definimos la clase **"Particula"** y le asignamos los atributos siguientes:

- __id = 0
- __origen_x = 0
- __origen_y = 0
- __destino_x = 0
- __destino_y = 0
- __veloicidad = 0
- __red = 0
- __green = 0
- __blue = 0
- __distancia = 0.0

Estos atributos tendrán un valor a través de su constructor, que por cierto, también ejecutará el siguiente paso. **Calcular la distancia euclidiana** en función a los parámetros dados en el constructor.

El cálculo de la distancia euclidiana no se hace directamente en el constructor, se hace a través del método **"distancia_euclidiana"** accesible desde el archivo **"algoritmos.py"**. Con este método le daremos un valor al atributo **distancia.**

```
""" Calculo de la distancia euclidiana """
self.__distancia = distancia_euclidiana(
    origen_x, origen_y, destino_x, destino_y)
```

El siguiente paso es generar el método **__str__** dentro de la clase **“Particula”** que nos permitirá imprimir los atributos de la clase dentro de un bucle.

Ahora generaremos una clase que administre múltiples instancias de la clase **“Particula”** que contenga los métodos **agregar_inicio, agregar_final, mostrar**.

```python
class listaParticula:
    def __init__(self):
        self.__particulas = []

    def agregar_inicio(self, particula: Particula):
        self.__particulas.insert(0, particula)

    def agregar_final(self, particula: Particula):
        self.__particulas.append(particula)

    def mostrar(self):
        for particula in self.__particulas:
            print(particula)
```

Dentro de cada método indicamos el segundo parámetro como instancia de la clase **“Particula”** para facilitar la lectura del código.

Finalmente generaremos 3 instancias de **“Partícula”** y las almacenaremos dentro de la lista. Este código esta dentro del archivo **“listaParticulas.py”.**

```python
""" id, origen_x, origen_y, destino_x, destino_y, veloicidad, red, green, blue, distancia """
particula1 = Particula(1, 5, 5, 10, 15, 3, 100, 100, 100, 0)
particula2 = Particula(2, 0, 10, 2, 12, 5, 200, 200, 200, 0)
particula3 = Particula(3, 5, 5, 10, 15, 3, 255, 100, 100, 0)

lista = listaParticula()

lista.agregar_final(particula2)
lista.agregar_final(particula3)
lista.agregar_inicio(particula1)

lista.mostrar()      You, 13 minutes ago • Uncommitted changes
```

Este es el orden, primero agregamos al final a *particula2* y después a *particula3*, al final agregamos a *particula1* pero al inicio de la lista. Después de todo el procedimiento, mostramos todos los elementos de la lista, sabremos que están ordenadas como indicamos por el primer parámetro del constructor, que es su *id*.

```
PS C:\Users\Armando\Documents\GitHub\Actividad 5> & C:/Users/Armando/AppData/Local/Programs/Python/Python310/python.exe "c:/Users/Armando/Documents/GitHub/Actividad 5/Actividad-5/src/listaParticulas.py"
####################################
Id: 1,
Origen X: 5,
Origen Y: 5,
Destino X: 10,
Destino Y: 15,
Velocidad: 3,
Rojo: 100,
Verde: 100,
Azul: 100,
Distancia: 11.180339887498949
####################################
Id: 2,
Origen X: 0,
Origen Y: 10,
Destino X: 2,
Destino Y: 12,
Velocidad: 2,
Rojo: 200,
Verde: 200,
Azul: 200,
Distancia: 2.8284271247461903
####################################
Id: 3,
Origen X: 7,
Origen Y: 5,
Destino X: 4,
Destino Y: 20,
Velocidad: 6,
Rojo: 255,
Verde: 100,
Azul: 100,
Distancia: 15.297058540778355
```

En esta ejecución logramos ver que ya están organizados como indicamos y además, la **distancia euclidiana** también esta calculada

Conclusiones

El desarrollo de la actividad fue bastante sencillo pero bastante retroalimentador para conocer el manejo de listas y clases de manera básico pero dentro del lenguaje Python.

Referencia

BOITES, M. D. (8 de 10 de 2020). Obtenido de Youtube.com: https://www.youtube.com/watch?v=KfQDtrrL2OU&ab_channel=MICHELDAVALOSBOITES

Código

Código de **"algoritmos.py"**

```python
import math


def distancia_euclidiana(x_1, y_1, x_2, y_2):
    """ Devuelve el resultado de la distancia euclidiana """
    """
        x_1 -- origen x
        x_2 -- destino x
        y_1 -- origen y
        y-2 -- destino y
     """
    return math.sqrt(pow(x_2 - x_1, 2) + pow(y_2-y_1, 2))
```

Código de **"Particulas.py"**

```python
from algoritmos import distancia_euclidiana


class Particula(object):
    __id = 0
    __origen_x = 0
    __origen_y = 0
    __destino_x = 0
    __destino_y = 0
    __veloicidad = 0
    __red = 0
    __green = 0
    __blue = 0
    __distancia = 0.0
```

```
    def __init__(self, id, origen_x, origen_y, destino_x, destino_y,
veloicidad, red, green, blue, distancia):
        """ Propiedades de la clase """
        self.__id = id
        self.__origen_x = origen_x
        self.__origen_y = origen_y
        self.__destino_x = destino_x
        self.__destino_y = destino_y
        self.__veloicidad = veloicidad
        self.__red = red
        self.__green = green
        self.__blue = blue
        self.__distancia = distancia
        """ Calculo de la distancia euclidiana """
        self.__distancia = distancia_euclidiana(
            origen_x, origen_y, destino_x, destino_y)

    def __str__(self):
        return (
            "#####################################\n"
            + "Id: " + str(self.__id) + ",\n"
            + "Origen X: " + str(self.__origen_x) + ",\n"
            + "Origen Y: " + str(self.__origen_y) + ",\n"
            + "Destino X: " + str(self.__destino_x) + ",\n"
            + "Destino Y: " + str(self.__destino_y) + ",\n"
            + "Velocidad: " + str(self.__veloicidad) + ",\n"
            + "Rojo: " + str(self.__red) + ",\n"
            + "Verde: " + str(self.__green) + ",\n"
            + "Azul: " + str(self.__blue) + ",\n"
            + "Distancia: " + str(self.__distancia))
```

Código de **“listaParticulas.py”**(en este archivo es donde inicia la ejecución)

```
from Particula import Particula


class listaParticula:
    def __init__(self):
        self.__particulas = []

    def agregar_inicio(self, particula: Particula):
        self.__particulas.insert(0, particula)

    def agregar_final(self, particula: Particula):
        self.__particulas.append(particula)
```

```
    def mostrar(self):
        for particula in self.__particulas:
            print(particula)


""" id, origen_x, origen_y, destino_x, destino_y, veloicidad, red, green,
blue, distancia """
particula1 = Particula(1, 5, 5, 10, 15, 3, 100, 100, 100, 0)
particula2 = Particula(2, 0, 10, 2, 12, 2, 200, 200, 200, 0)
particula3 = Particula(3, 7, 5, 4, 20, 6, 255, 100, 100, 0)

lista = listaParticula()

lista.agregar_final(particula2)
lista.agregar_final(particula3)
lista.agregar_inicio(particula1)

lista.mostrar()
```

Subida al git

```
Armando@Armando04 MINGW64 ~
$ cd "C:\Users\Armando\Documents\GitHub\Actividad 4"

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 4 (master)
$ git add .

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 4 (master)
$ git add *

Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 4 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   __pycache__/mainwindow.cpython-310.pyc
        new file:   __pycache__/ui_mainwindow.cpython-310.pyc
        new file:   main.py
        new file:   mainwindow.py
        new file:   mainwindow.ui
        new file:   ui_mainwindow.py


Armando@Armando04 MINGW64 ~/Documents/GitHub/Actividad 4 (master)
$ git commit -m "Primer commit"
[master (root-commit) 4224592] Primer commit
 6 files changed, 254 insertions(+)
 create mode 100644 __pycache__/mainwindow.cpython-310.pyc
 create mode 100644 __pycache__/ui_mainwindow.cpython-310.pyc
 create mode 100644 main.py
 create mode 100644 mainwindow.py
 create mode 100644 mainwindow.ui
 create mode 100644 ui_mainwindow.py
```