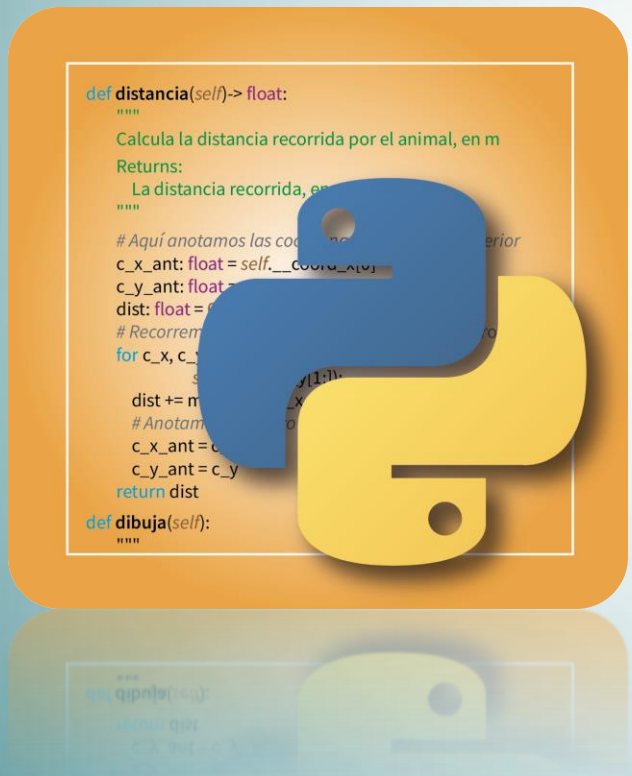
Programación

Práctica 1. Piscifactoría



Michael González Harbour

José Javier Gutiérrez García

José Carlos Palencia Gutiérrez

José Ignacio Espeso Martínez

Adolfo Garandal Martín

Departamento de Ingeniería Informática y Electrónica

Práctica 1: piscifactoría

Objetivos: Practicar con expresiones para hacer cálculos y con la invocación de funciones

Descripción: Se desea hacer una aplicación que nos permita realizar cálculos del crecimiento esperado de peces en una piscifactoría

Notación:

- p : peso del pez en gramos
- p_0: peso inicial del pez, en gramos
- d : tiempo transcurrido, en días
- $T(d)$: suma de las temperaturas medias diarias de los días de crecimiento, en ºC
- CCT: coeficiente de crecimiento térmico, en gramos$^{1/3}$/(dia*ºC)
- K: factor de crecimiento, en gr$^{1/3}$

Cálculo del crecimiento y módulos

El cálculo del peso se hace con la siguiente ecuación

$$p = (p_0^{1/3} + CCT \cdot T(d) + K)^3$$

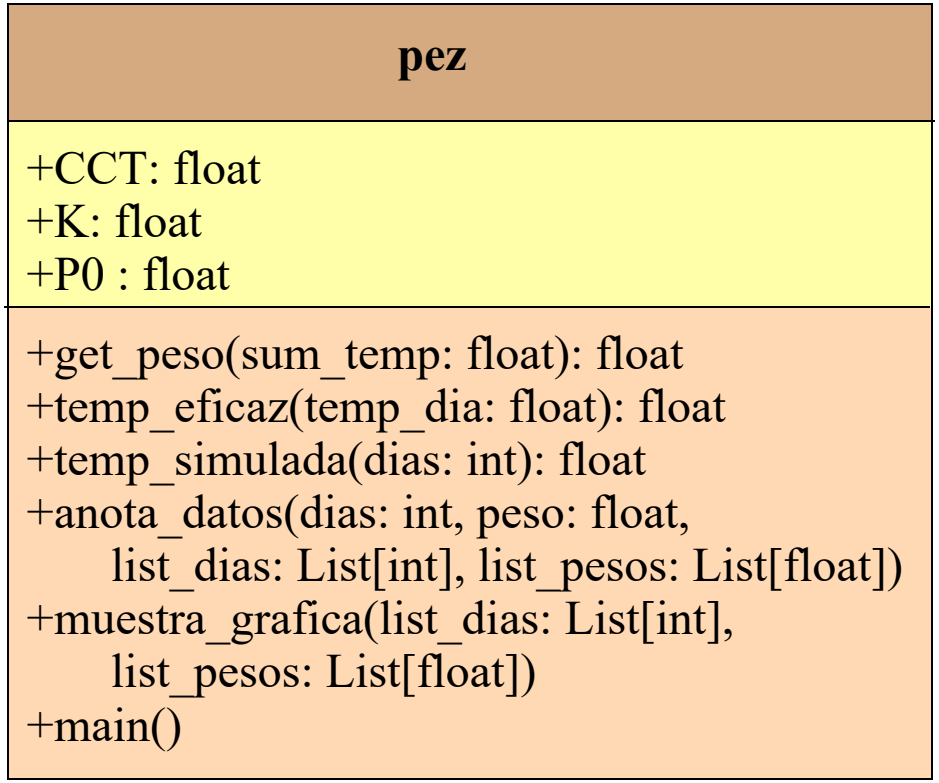
Diseñaremos el software con un módulo llamado `pez.py` con funciones que hacen los cálculos de crecimiento del pez y un `main`

Fuente: Modelo aleatorio de crecimiento CCT biparamétrico, Alamar, M.; Estruch, V.; Pastor, J. y Vidal, A., http://www.revistas.ieo.es/index.php/boletin_ieo/article/view/157

Módulo pez.py

Contiene las siguientes constantes:

- coeficientes de crecimiento (CCT y K): `CCT`, `K`
- peso inicial (p_0): `P0`
  - para una lubina típica, CCT= 0.000897653, K=0.116375, P0=38.97



También contiene estas funciones:

- `get_peso`(), que calcula y retorna el peso actual del pez, usando la fórmula del peso
  - recibe como parámetro `sum_temp`, que es el valor $T(d)$ de la fórmula
- `temp_eficaz()`: retorna la temperatura eficaz usada para el cálculo del crecimiento

Módulo `pez.py` (cont.)

- `temp_simulada()`: retorna la temperatura simulada en el día indicado
- `anota_datos()`: Añade los valores `dias` y `peso` a las listas de dias y pesos
- `muestra_grafica()`: Muestra la gráfica del peso, dadas las listas de valores de días y pesos

De estas funciones solo se pide completar `get_peso()`

Programa principal

También se pide crear en el módulo `pez.py` una función `main()`, que deberá realizar la simulación del crecimiento de una lubina y mostrar un gráfico

Para ello debe realizar los siguientes pasos:

- crear dos listas vacías para los dias y los pesos
  - esto se da hecho
- crear la variable `dias` para el número de dias transcurridos
  - valor inicial 0
- crear la variable `sum_temp` para la suma de las temperaturas medias diarias
  - valor inicial 0

Programa principal (cont.)

- bucle que se repite durante 270 dias; este bucle se da ya hecho, y en su interior hay que hacer los siguientes pasos
  - sumar uno a `dias`
  - `temp` = obtener la temperatura simulada usando como parámetro la variable `dias`
  - `eficaz` = obtener la temperatura eficaz usando como parámetro la variable `temp`
  - añadir `eficaz` a `sum_temp`
  - `peso` = obtener el peso usando como parámetro la variable `sum_temp`
  - anotar los datos `dias` y `peso` en `list_dias` y `list_pesos`, con `anota_datos()`
- finalizado el bucle, invocar a `muestra_grafica()` usando como parámetros las variables `list_dias` y `list_pesos`

Parte avanzada

Crear una función similar a `temp_simulada()` en la que las temperaturas sean fijas (no aleatorias) y obtenidas de la siguiente tabla:

Días	Temp. media (ºC)
1 a 92	23.0
93 a 155	25.0
156 a 207	26.0
otros valores	29.0

Modificar el `main()` para que:

- use la nueva función en lugar de `temp_simulada()`
- tras mostrar la gráfica, añadir un día a 31 ºC y comprobar mostrándolo en la pantalla que el peso del pez sale con valor `nan` (pez muerto)

Entregar dos ficheros

1. El código fuente Pyhton
2. Un ***informe*** conteniendo:
   - una captura de pantalla de la gráfica
   - además, si se ha hecho la parte avanzada: una captura de pantalla de la nueva gráfica y los resultados de la ejecución del `main`, mostrando el pez muerto