Examen

Febrero

2 de Febrero 2015

Informática
Año 2014/2015
Facultad de CC.
Matemáticas

▷ **1. Elegir los tres mayores cuadrados perfectos.**

(3 puntos) Implementa una función `max3cuadradosPerfectos` que, dada una lista de números enteros, devuelva los 3 números mayores de esta lista que sean cuadrados perfectos. En el caso de que alguno de estos números no exista, por ejemplo si únicamente existe un cuadrado perfecto en la lista, se deberá devolver el valor -1 en el lugar de los que no existen.

Se valorará la eficiencia de la solución propuesta, entendiendo como una solución más eficiente que la función use un único bucle y no utilice estructuras de datos adicionales. Se permite, y recomienda, el uso de funciones adicionales para resolver el ejercicio.

Por ejemplo, la función se debe comportar como se indica a continuación:

```
>>> max3cuadradosPerfectos ([2,4,5,3,7,8])
(4, -1, -1)

>>> max3cuadradosPerfectos ([64,4,10,6,7,100, 10000])
(10000, 100, 64)

>>> max3cuadradosPerfectos ([7,3,532,1002, 2, 99])
(-1, -1, -1)
```

▷ **2. Aproximación de π**

Considera las sucesiones a_n e y_n definidas a continuación:

- Empezamos con

$$\begin{aligned} a_0 &= 6 - 4\sqrt{2} \\ y_0 &= \sqrt{2} - 1 \end{aligned}$$

- Y luego seguimos

$$y_{k+1} = \frac{1 - (1 - y_k^4)^{1/4}}{1 + (1 - y_k^4)^{1/4}}$$

$$a_{k+1} = a_k(1 + y_{k+1})^4 - 2^{2k+3} y_{k+1}(1 + y_{k+1} + y_{k+1}^2)$$

Resulta que la sucesión a_n converge a $1/\pi$.

- **(1.5 puntos)** Escribe una función en python `aprox_pi(k)` que use el término k de la sucesión a_n para calcular π
- **(1.5 puntos)** Escribe una función en python `aprox_pi(epsilon)` usando la misma idea para calcular π pero usando el término k de la sucesión a_n tal que $abs(a_k - a_{k-1}) < \epsilon$

▷ 3. Marcos de fotos.

Una importante empresa dedicada a la fotografía ha solicitado el desarrollo de un algoritmo para poner marcos a las fotografías digitalmente. Para simplificar el problema, trataremos únicamente con fotografías en blanco y negro. Este marco consiste en añadir un borde de k píxeles al exterior de una imagen. De esta forma, se deberá generar, a partir de una imagen inicial, una imagen que contenga tanto el marco pedido como la imagen original.

- **(2 puntos)** Escribe una función `poner_marco(img,k,color)` que devuelva la imagen `img` con el correspondiente marco de color `color`.
- **(2 puntos)** Mejorar la función anterior para poder pintar los lados horizontales del marco de un color cualquiera y los verticales de otro.

Pista:

- El tamaño de una imagen `img` se puede calcular de la siguiente forma:

  ```
  ancho, alto = img.size
  ```

  donde `ancho` es el número de pixels horizontales que tiene la imagen y `alto` es el número de pixels verticales.

- Para generar una imagen nueva, se debe hacer uso de la función `new`. Por ejemplo, si queremos generar la imagen en una variable de nombre `imgMarco`, pondremos:

  ```
  imgMarco = Image.new("L",(anchoNew, altoNew), "white")
  ```

  donde `anchoNew` es el número de píxeles horizontales que tendrá la imagen y `altoNew` es el número de píxeles verticales.

- Para escribir un pixel en una coordenada de una imagen se debe utilizar la función `putpixel`. Así la instrucción

  ```
  img.putpixel ((x,y), 0)
  ```

  pondría un pixel negro en las coordenada `(x,y)` de la imagen `img`.

- Una imagen y su transformada por la función *mejorada*