Lenguajes de Inteligencia Artificial

Segundo curso. Primer cuatrimestre
Ingeniería Técnica en Informática de Gestión
e Ingeniería Técnica en Informática de Sistemas
Escuela Politécnica Superior
Universidad de Córdoba
Curso académico: 2009 - 2010

Práctica número 3: Iteración y recursión

1. **(*) Sucesión de Fibonacci y el número áureo**
   - Codifica una función denominada ***fibonacci*** que utilice la fórmula de Binet

$$Fibonacci(n) = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right]$$

   para calcular el término n-ésimo de la sucesión de Fibonacci: 1, 1, 2, 3, 5, 8, etc.

   - Codifica una función **iterativa** denominada ***numero_aureo*** que
     - Reciba como parámetro una cota de error
     - Permita calcular una aproximación al límite de la siguiente sucesión numérica

$$\boldsymbol{a_n = f_{n+1} / f_n}$$

     donde $\boldsymbol{f_n}$ representa el término n-ésimo de la sucesión de Fibonacci

     - La función debe finalizar cuando dos elementos consecutivos de la sucesión $\boldsymbol{a_n}$ y $\boldsymbol{a_{n+1}}$ disten menos que el valor introducido como cota de error

2. **(*) Límite de un sucesión convergente**
   - Codifica una función **iterativa** denominada **"límite"** que permita calcular una aproximación al límite de una sucesión numérica convergente.
   - La función debe recibir como argumentos a:
     - La **cota de error**, que permitirá terminar la función cuando dos elementos consecutivos de la sucesión disten menos que dicha cota de error.
     - Una **función** que represente el término general de la sucesión.

   - ¿Cómo se llamaría a la función "límite" si se desea calcular el límite de la sucesión numérica cuyo término general es

$$a_n = (1 + 1/n)^n$$

   con una cota de error de 0.001?

3. **(*)** Aproximaciones al **número π:**
   - **Leibniz** propuso la siguiente serie numérica para calcular una aproximación a $\pi/4$:

$$\frac{\pi}{4} = \sum_{1}^{\infty} f(n) = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots$$

     a. Escribe una función denominada **término-Leibniz** que reciba un número entero **n** y calcule el "n-ésimo" término de la serie de Leibniz.
     b. Escribe una función **iterativa,** denominada **Leibniz-pi-1**, que reciba como parámetro el número de términos **n** de la serie propuesta por Leibniz que se deseen sumar.
     c. Escribe una función **iterativa,** denominada **Leibniz-pi-2**, que reciba como parámetro una **cota de error** y termine cuando la diferencia entre dos términos consecutivos de la sucesión sea inferior a dicha cota.

   - **Wallis** propuso utilizar la siguiente serie para calcular una aproximación a π**/4**:

$$\frac{\pi}{4} = \frac{2}{3}\times\frac{4}{3}\times\frac{4}{5}\times\frac{6}{5}\times\frac{6}{7}\times\frac{8}{7}\times\frac{8}{9}\times\ldots$$

a. Codifica una función denominada **factor-Wallis** que reciba como parámetro un número natural **n** y devuelva como resultado el n-ésimo factor de la sucesión de Wallis.

   Por ejemplo:

   (factor-Wallis 1) ==> 2/3
   (factor-Wallis 2) ==> 4/3
   …
   (factor-Wallis 5) ==> 6 /7

b. Escribe una función **iterativa** denominada **Wallis-iterativa** que reciba como parámetro un **número** natural que indicará cuántos factores se han de multiplicar.
c. Escribe función **recursiva de cola** denominada **Wallis-recursiva** que reciba como parámetro una cota de error, de forma que la función terminará su ejecución cuando el factor que se vaya a multiplicar esté comprendido entre los siguientes valores:

   1 - cota < factor < 1 + cota

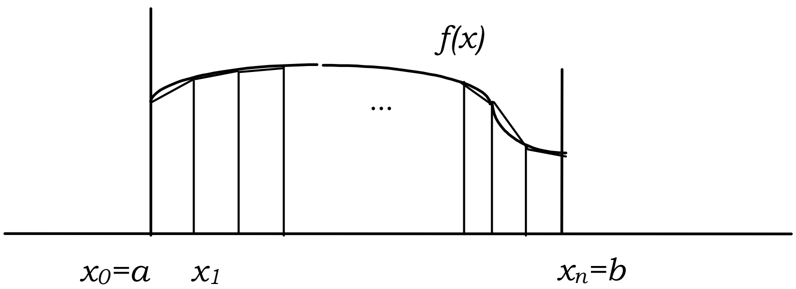
- **Observación:** la sucesión de Wallis converge "muy lentamente".

4. (*) Codifica una función **iterativa**, denominada **integral**, que reciba cuatro parámetros:
   - Los dos extremos de un intervalo: ***a*** y ***b***
   - Una función ***f*** que sea positiva en el intervalo **[a,b]**
   - Un número "n"

   y devuelva la aproximación a la integral según el **método de los trapecios**.

$$\int_a^b f(x)\,dx = \sum_{i=0}^{n-1}\left(\frac{f(x_i)+f(x_{i+1})}{2}\right) * h$$

donde h = (b - a) / n y $x_i = a + i * h$



- ¿Cómo se calcularía el área de la función f(x) = $3x^2$ +1 definida en el intervalo [1,3]?

5. **Simpson** propuso un método para calcular la aproximación a la integral de una función en un intervalo [a,b]:

$$\int_a^b f(x) = \left(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + \ldots + 2y_{n-2} + 4y_{n-1} + y_n\right)\frac{h}{3}$$

donde

- *n* es un número es par

- $h = (b - a) / n$
- $y_k = f(a + h\,k)$

a. Codifica una función denominada **término-Simpson** que calcule el k-ésimo término de la sucesión de Simpson a partir de los siguientes parámetros:
   - La función **f**
   - El extremo inferior del intervalo **a**
   - El incremento **h**
   - El número natural **k**

b. Utiliza la función **término-Simpson** para codificar una función **iterativa** denominada **Simpson-iterativo** que calcule la aproximación a la integral de una función **f** en un intervalo **[a, b]** utilizando **n** términos.

c. ¿Qué valor se obtiene al calcular el área de la función $f(x) = 3\,x^2$ en el intervalo $[0,1]$ si se utiliza el valor $n = 100$?

6. Codifica una función **iterativa** denominada **“suma-serie-numérica”** que permita calcular una aproximación a la suma de una serie numérica. La función debe recibir los siguientes parámetros:
   - La **cota de error**, que permitirá terminar la función cuando un elemento de la serie sea menor que dicha cota de error.
   - Una **función** que represente el término general de la sucesión.

   a. ¿Cómo se llamaría a la función “suma-serie-numérica” si se desea calcular la suma de la serie cuyo término general es $a_n = 1/n^2$ con una cota de error de 0.001?

7. Codifica la siguiente función de Ackerman:

$$
\begin{array}{ll}
A(0,y) = 1 & \forall y \geq 0 \\
A(1,0) = 2 & \\
A(x,0) = x + 2 & \forall x \geq 2 \\
A(x,y) = A(A(x-1, y), y-1) & \forall x, y \geq 1
\end{array}
$$

8. Codifica una función denominada ***incremento-funcional*** que reciba una función ***f*** como parámetro y devuelva como resultado la función que calcularía la siguiente expresión

$$\frac{f(x+1) - 2f(x) + f(x-1)}{4}$$

¿Cómo se invocaría la función ***incremento-funcional***?

9. **(*)** Codifica una función denominada ***diferencia-simétrica*** que reciba como parámetros dos funciones ***f*** y ***g*** y devuelva como resultado la función que calcularía la siguiente expresión:

$$|f(x) - g(x)|$$

¿Cómo se invocaría la función ***diferencia-simétrica***?

10. **(*)** El algoritmo de ***Euclides*** permite calcular el máximo común divisor (M.C.D.) de dos números naturales:

Dados “a” y “b”, dos números naturales, donde "a >= b",
si a = c b + r entonces M.C.D.(a, b) = M.C.D.(b,r).

- *El algoritmo* concluirá *cuando el segundo argumento sea cero, siendo el máximo común divisor el primer argumento. Si “a” es menor que “b”, se calculará el*

M.C.D.(b,a).

V.g.: Cálculo del máximo común divisor de 630 y 198

a	**630**	198	36	**18**
b	**198**	36	18	0
r	36	18	0	

M.C.D.(630,198) = 18

a. Codifica una función **iterativa**, denominada **mcd-iterativo**, que permita calcular el máximo común divisor de dos números.
b. Codifica una función **recursiva**, denominada **mcd-recursivo**, que permita calcular el máximo común divisor de dos números.

11. **(*)** Dos números naturales son **amigos** si la suma de los divisores de uno es igual al otro número y viceversa.
    - El menor par de números amigos es el formado por el 220 y 284:
        - Suma de los divisores de 220 (excepto 220):
          1 + 2 + 4 + 5 + 10 + 20 + 11 + 22 + 44 + 55 + 110 = 284
        - Suma de los divisores de 284 (excepto 284):
          1 + 2 + 4 + 71 + 142 = 220
    - Otros números amigos son (1184 y 1210) (6232 y 6368), (2620 y 2924)...

    a. Codifica una función **recursiva** denominada **suma-divisores** para calcular la suma de los divisores de un número natural (excepto el propio número)
    b. Utiliza la función **suma-divisores** para codificar un predicado denominado **amigos?** que permita comprobar si dos números son o no amigos.
    c. Utiliza el predicado **amigos?** para codificar un predicado denominado **perfecto?** que permita comprobar si un número es perfecto, es decir, es igual a la suma de sus divisores inferiores a él.

    Por ejemplo: el número 28 es perfecto porque

    28 = 14 + 7 + 4 + 2 + 1

    - Nota: si se utiliza el par (28 y 28) y se comprueba que son "amigos" entonces 28 será "perfecto".

12. **(*)** Un número es **primo** si no tiene divisores propios menores que su raíz cuadrada. Codifica dos predicados que determinen si un número es primo no:
    a. El primer predicado se denominará **primo-iterativo?** y utilizará la forma especial "**do**" para crear una función iterativa.
    b. El segundo predicado se denominará **primo-recursivo?** y será una función recursiva.