Tarea 2
Algoritmos y Complejidad

«Horner y Fibonacci»

Algorithm Knaves

2021-11-07

El método de Horner para evaluar polinomios se basa en escribir:

$$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0 = ((\cdots(a_n x + a_{n-1})x + a_{n-2})x + \cdots)x + a_0$$

Es la manera más eficiente de evaluar polinomios en general. Una simple extensión permite calcular también la derivada del polinomio, algoritmo 1. Al final p es el valor

Algoritmo 1: El método de Horner

$p \leftarrow a_n; q \leftarrow 0$
for $i \leftarrow n-1$ **downto** 0 **do**
  $q \leftarrow p + x \cdot q$
  $p \leftarrow a_i + x \cdot p$
end

del polinomio en x, q es su derivada en ese mismo punto.

1. Escriba una función Python llamada horner que tome una tupla de coeficientes a de un polinomio y un valor x y retorne una tupla de dos elementos conteniendo el valor del polinomio con coeficientes a y su derivada en x. Debe ceñirse a la declaración:

```
def horner(a: List[float],  # Coefficients
           x: float         # Value of x
          ) -> Tuple[float, float]:
                           # (p(x), p'(x))
```

(30 puntos)

2. Escriba una función Python llamada newton que tome la lista de coeficientes a de un polinomio, un valor inicial x0 y una tolerancia eps y retorne una aproximación a un cero del polinomio obtenida usando el método de Newton partiendo del valor inicial x0. Itere hasta que valores sucesivos difieren en menos

de eps. Debe usar la función de la pregunta 1, solo en caso que no responda esa pregunta puede usar la función not_horner del listado 1.

```
def not_horner(a: List[float],  # Coefficients
               x: float         # Value of x
              ) -> Tuple[float, float]:
                  # (p(x), p'(x))
    p = 0.0
    for i in range(len(a)):
        p += a[i] * x**i

    q = 0.0
    for i in range(1, len(a)):
        q += i * a[i] * x**(i - 1)
    return (p, q)
```

Listing 1: La función not_horner

Su función debe ceñirse a la declaración:

```
def newton(a: List[float], # Coefficients
           x0: float,      # Initial value
           eps: float      # Tolerance
          ) -> float:      # Zero found
```

(30 puntos)

Los números de Tribonacci se definen mediante la recurrencia:

$$T_{n+3} = T_{n+2} + T_{n+1} + T_n \qquad T_0 = T_1 = 0, T_2 = 1$$

En forma afín se definen los números de N-bonacci:

$$A_{n+N} = \sum_{0 \le r \le N} A_{n+r} \qquad A_0 = A_1 = \ldots = A_{N-2} = 0, A_{N-1} = 1$$

Técnicas estándar de solución de recurrencias muestran que si ρ_N es el único cero positivo del polinomio:

$$x^N - \sum_{0 \le r \le N-1} x^r$$

entonces:

$$\lim_{n \to \infty} \frac{A_{n+1}}{A_n} = \rho_N$$

3. Demuestre que $1 < \rho_N < 2$.

(5 puntos)

4. Use su programa de la pregunta 2 para obtener ρ_5 (para los números de 5-bonacci) con cinco cifras. ¿Qué valor inicial sugieren las cotas de 3?

   (15 puntos)

5. Verifique el límite citado evaluando la fracción para algunos valores de n.

   (15 puntos)

Asegúrese de adjuntar los programas que escribe.

Condiciones de entrega

- La tarea se realizará *individualmente* (esto es grupos de una persona), sin excepciones.
- La entrega debe realizarse vía Aula en un *tarball* en el área designada al efecto, en el formato `tarea-2-`*`rol`*`.tar.gz` (`rol` con dígito verificador y sin guión).

  Dicho *tarball* debe contener las fuentes en LaTeX 2$_\varepsilon$ (al menos `tarea-2.tex`) de la parte escrita de su entrega, además de un archivo `tarea-2.pdf`, correspondiente a la compilación de esas fuentes.
- Asegúrese que todas sus entregas tengan sus datos completos: número de la tarea, ramo, semestre, nombre y rol. Puede incluirlas como comentarios en sus fuentes LaTeX (en TeX comentarios son desde % hasta el final de la línea) o en posibles programas. Anótese como autor de los textos.
- En la portada de su texto deberá incluir una tabla como la siguiente:

Concepto	**Tiempo [min]**
Revisión	
Desarrollo	
Informe	

  Acá *revisión* incluye revisión de apuntes, búsquedas en Internet, lectura de otras referencias; *desarrollo* es el tiempo invertido en la solución pedida; *informe* se refiere al tiempo requerido para confeccionar los entregables.
- Si usa material adicional al discutido en clases, detállelo. Agregue información suficiente para ubicar ese material (en caso de no tratarse de discusiones con compañeros de curso u otras personas).
- Su programa ejecutable *debe* llamarse `tarea2`, de haber varias preguntas solicitando programas, estos deben llamarse usando el número de la pregunta, como `tarea2-1`, `tarea2-2`, etc. Si hay programas compilados, incluya una `Makefile` que efectúe las compilaciones correspondientes.

  Los programas se evalúan según que tan claros (bien escritos) son, si se compilan y ejecutan sin errores o advertencias según corresponda. Parte del puntaje es por ejecución correcta con casos de prueba. Si el programa no se ciñe a los requerimientos de entrada y salida, la nota respectiva es cero.

- La entrega debe realizarse dentro del plazo indicado en Aula.
- Nos reservamos el derecho de llamar a interrogación sobre algunas de las tareas entregadas. En tal caso, la nota base (antes de descuentos por atraso y otros) es la de la interrogación. No presentarse a la interrogación sin justificación previa significa automáticamente nota cero.