Introducción a la Computación / Práctico 8

Definiciones Inductivas

1. Conjuntos definidos por inducción:
    a. Considere el conjunto R de los números reales. Dé ejemplos de conjuntos A ⊂ R que satisfagan los siguientes conjuntos de cláusulas:
        - 0 ∈ A,
          Si n ∈ A entonces (n+1) ∈ A.
        - 2 en A,
          Si n ∈ A entonces (n+2) ∈ A.
        - 3 ∈ A,
          Si n ∈ A entonces (n+1) ∈ A,
          Si n ∈ A entonces (n-1) ∈ A.
        - Si (n+1) ∈ A entonces n ∈ A.
    b. Para cada conjunto de cláusulas, indique cuál es el mínimo subconjunto de R que satisface las cláusulas.
    c. Para cada uno de los conjuntos mínimos, dé 3 elementos que pertenezcan al conjunto, de ser posible, y justifique su pertenencia en términos de la aplicación de las cláusulas
2. Defina inductivamente los siguientes conjuntos:
    a. los naturales múltiplos de 3 (o sea {0, 3,6,9, ...}).
    b. los enteros múltiplos de 3.
    c. los naturales que son potencias de 2, (o sea {1, 2, 4, 8, 16, ...}).
    d. { (x, y) / x∈N, y∈N, x<y }
    e. { (x, y) / x∈N, y∈N, x≤y }
    f. { (x, x!) / x∈N }
    g. { (x, y) / x∈N, y = Σ i (es la suma de i=0 hasta x de i, la suma de los primeros x números naturales)}
3. Defina inductivamente los siguientes conjuntos:
    a. $\{a\}^* = \{a^n / n \geq 0\} = \{e, a, aa, aaa, aaaa, ...\}$
    b. $\{b\}+ = \{b\}^* - \{e\} = \{b^n / n>0\}$
    c. $\{a^n b^n / n \geq 0\} = \{e, ab, aabb, aaabbb, aaaabbbb, ...\}$
    d. $\{(ab)^n / n \geq 0\} = \{e, ab, abab, ababab, abababab, ...\}$
    e. $\{a^n b^m / n \geq 0, m \geq 0$ y $n+m$ es par$\}$
    f. $\{a^n c b^{2n+1} / n \geq 0\}$
    g. $\{a^n b^m / m > n \geq 0\}$
    h. El lenguaje de las cadenas binarias (con símbolos 0 y 1) que representan números pares. La cadena vacía (e) no pertenece al lenguaje.
    i. El lenguaje de las cadenas binarias que empiezan con el símbolo 1.
    j. El lenguaje de las cadenas de elementos de un conjunto S que no poseen elementos consecutivos repetidos. La cadena vacía (e) pertenece al lenguaje. Puede usar la igualdad de elementos de S.
    k. El lenguaje de las cadenas ordenadas de menor a mayor de elementos de un conjunto S. Asuma una relación de orden total £ entre los elementos de S.

Pruebas por Inducción

4. Sea S = {a,b} y considere D ⊂ S* definido inductivamente por las cláusulas:
    - a ∈ D

   - Si a ∈ D entonces bab ∈ D.

   Enuncie el principio de inducción primitiva para D y pruebe inductivamente que para todo a ∈ D, a tiene un número par de símbolos b.
5. Sea S = {a,b,c} y considere D ⊂ S* definido inductivamente por las cláusulas:
   - e ∈ D
   - Si x ∈ D entonces bxbc ∈ D
   - Si x ∈ D y entonces bxba ∈ D

   Dé 3 palabras (cadenas) de S* que pertenezcan a D y 3 palabras que no pertenezcan. Sea long(a) la longitud de la palabra a. Cuáles de las siguientes afirmaciones son correctas? Justifique formalmente su respuesta:
   a. Si a ∈ D entonces long(a) > 0.
   b. Si a ∈ D entonces long(a) es par.
   c. Si a ∈ D entonces long(a) es múltiplo de 3.
   d. Si a ∈ D entonces a tiene el mismo número de ocurrencias de "a" que de "c".
   e. Si a ∈ D entonces a tiene al menos el doble de símbolos "b" que de símbolos "a".
6. Sea S = {a,b,c}. Sea D ⊂ S* definido inductivamente por las cláusulas:
   - e ∈ D
   - Si x ∈ D entonces xb ∈ D
   - Si x ∈ D entonces xa ∈ D

   Considere G ⊂ S* definido inductivamente por las cláusulas:
   - e ∈ G
   - Si x ∈ G entonces bx ∈ G
   - Si x ∈ G entonces ax ∈ G

   Cuáles de las siguientes afirmaciones son correctas?. Justifique formalmente.
   - D ⊂ G
   - G ⊂ D
   - D = G

Funciones Recursivas

7. Considere la siguiente función f: N → N definida por recursión primitiva en N:
   f(0) = 0
   f(n+1) = f(n)+2(n+1)
   a. Calcule f(3), f(5) y f(7) aplicando paso a paso la definición de f.
   b. Pruebe por inducción en n que f(n) = n.(n+1), para todo n de N.
8. Defina por recursión en n la función f: N → N que calcula f(n) = $\Sigma_{k=0}^{n}$(k.k!) (nota: esto es la suma de k=0 hasta n de (k.k!)).
   Pruebe por inducción en n que f(n) = (n+1)!-1.
9. ¿Qué calcula la siguiente función? f : NxN → N
   f(0,n) = 0
   f(n+1,0) = 0
   f(n+1,m+1) = 1+f(n,m)
10. Considere la siguiente definición de g : NxN → Z
    g(n,0) = n
    g(n, m+1) = g(n,m)-1
    a. Calcule g(5,2) y g(2,5)
    b. Demuestre por inducción que para todo n,m ∈ N, si n ≥ m entonces g(n,m) ≥ 0 y g(n,m) = n-m.
11. Defina la función mod: N x N → N que calcula el resto de la división entera (positiva) entre dos números naturales. Ejemplos: mod(5,2) = 1, mod(8,2) = 0, mod(1,2) = 1. Asuma como precondición que el segundo argumento es mayor que cero. Utilice únicamente la

operación de resta entre números naturales (no puede usarse la operación div).

12. Defina la función esPotenciaDe2: N → {0,1} que dado un número natural n, esPotenciaDe2(n) = 0 si y sólo si n es potencia de 2.
13. Sea S ={a,b,c,d,...,x,y,z}. Defina recursivamente la función par: S* → {0,1} tal que par(a) es 1 si y sólo si a tiene un número par de letras.
14. Sea S un alfabeto. Defina la función duplicar: S* → S* que dada una palabra a calcula la palabra resultante de duplicar cada letra de a. Por ejemplo, duplicar(abc) = aabbcc.
15. Sea S un alfabeto. Defina la función copiar: N x S* → S* que dada una palabra a de S* y un número n, arma una palabra compuesta de n copias de a. Por ejemplo, copiar(3,ab) = ababab.
16. Sea S un alfabeto.
    a. Defina una función eliminar: S* x S → S* que, dada una palabra a y un caracter x, retorna la palabra a sin la primera ocurrencia de x, si hay al menos una. Por ejemplo, eliminar(abcbef,b) = acbef y eliminar(abcbef,d) = abcbef.
    b. Defina una función eliminarTodas: S* x S → S* que, dada una palabra a y un caracter x, retorna la palabra a sin las ocurrencias de x. Por ejemplo, eliminarTodas(abcbef,b) = acef y eliminarTodas(abcbef,d) = abcbef.
17. Defina recursivamente las funciones:
    p(a,b) = a*b ; m(a,b)= a mod b ; c(a,b)= a div b; mcd(a,b).
18. Encontrar las funciones que definen las siguientes recurrencias:
    a. q(0)=0 , y si n >=1 q(n)=q(n-1)+2n - 1
    b. r(0)=1, y si n>=1 r(n)=2r(n-1)
    c. s(0)=1, y si n>=1 s(n)=s(s(n-1)-1)+1

    Nota: Se recomienda utilizar expansión de recurrencias
19. Mostrar que el enésimo número de Fibonacci fib(N) puede calcularse como f(0,1,n) definiendo f(a,b,0) = a y si c > 0, f(a,b,c) = f(b,a+b,c-1).

Procedimientos Recursivos

20. Implementar recursivamente el cálculo de los números de Fibonacci. Los números de Fibonacci constituyen una serie donde el de orden 0 es 0, el de orden 1 es 1 y los demás son la suma de los dos anteriores. ¿Cuántas veces se calcula fib(i) en el cálculo de fib(j) con j > i? Realizar el diagrama de llamadas para fib(n).
21. Diseñar una rutina recursiva listar1an(n) que liste los números del 1 al n. ¿Podría lograr que los liste en orden inverso? ¿Cómo compara esta implementación con la equivalente y más obvia en base a una estructura de control repetitiva?
22. Diseñar una rutina recursiva promedio(VEC,N) que retorne el promedio de los valores del array VEC de N elementos. Una forma de determinar la suma de N elementos es iterar sobre ellos calculándola. Otra forma es partir el problema, por ejemplo en dos. Sumar la parte izquierda, sumar la parte derecha y calcular la suma total como la suma de la parte izquierda más la suma de la parte derecha.
23. Defina una función recursiva int busco(VEC,izq,der N,dato) que retorna la posición ocupada por dato sobre el array VEC entre las posiciones izq y der de vec o -1 si no se encuentra el dato.
    a. Una forma de resolver el problema podría ser partir el problema en un subproblema izquierdo y uno derecho y buscar en la parte derecha y en la parte izquierda. Resuelva el problema de esta manera.
    b. ¿Ayudaría que los elementos de VEC se encontraran ordenados?
    c. Implemente una versión mejorada en el supuesto de que los elementos de VEC se encuentran ordenados.
24. Desarrollar una función recursiva que reciba un array y retorne su mínimo elemento.
25. Idem anterior pero que retorne la posición del mínimo.

26. Usando el ejercicio anterior implemente una función recursiva que ordene un array.
27. Considere la siguiente representación de polinomios mediante Listas de coeficientes:
    $P(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x^1 + a_0 x^0$ se representa mediante la lista $a_n, a_{n-1}, \ldots, a_1, a_0$
    Escribir una función recursiva para evaluar un polinomio usando la regla de Horner ($P(x) = ((\ldots((a_n * x) + a_{n-1})x) + \ldots + a_1)x + a_0$).
28. Diseñar una rutina inserto(VEC,N,CLAVE) que inserta CLAVE en el vector ordenado VEC de 100 elementos. El vector es de 100 elementos, pero contiene datos hasta el lugar N.
29. Eratóstenes construyó una criba para determinar los números primos del 2 al 100 de la siguiente manera:
    Sobre una madera escribió los números.
    Recorría la misma secuencialmente buscando el primer número no perforado. Al encontrarlo, perforaba todos sus múltiplos y repetía a partir del siguiente, hasta terminar.
    Implemente una versión estrictamente recursiva (sin usar estructuras de control repetitivas) de este algoritmo.
30. Diseñar una función recursiva booleana primo(n) que devuelve true o false según su parámetro sea primo o no.
31. Problema de las Torres de Hanoi:
    a. Escribir un procedimiento que resuelva el problema llamado de las "Torres de Hanoi". Existen tres cilindros verticales donde es posible insertar discos. En uno de los cilindros hay *n* discos todos de diferente tamaño, colocados en orden de tamaño con el más chico arriba. Los otros dos cilindros están vacíos.
       El problema es pasar la torre de discos a otro de los cilindros usando como único movimiento elemental el cambio de un disco de un cilindro a otro cualquiera, sin que en ningún momento un disco vaya a colocarse encima de otro más chico que él.
    b. Escribir procedimientos de modo de imprimir secuencias de movimientos elementales que resuelvan el problema de las torres de Hanoi de cualquier tamaño.
    c. Una antigua leyenda cuenta que en un monasterio de Hanoi, los monjes disponen de 64 discos perforados de diferente tamaño, que superpuestos en orden descendente de tamaño, conforman una torre. Tienen, además tres ejes uno de origen, otro de destino y un tercer eje que puede ser utilizado como auxiliar. El problema que los monjes pretenden resolver es encontrar la secuencia de movimiento de discos que permita trasladar la torre del eje origen al destino con la condición de que en ningún momento se apoye un disco sobre uno de menor tamaño. Los monjes realizan un movimiento por día y esperan que cuando concluyan se acabe el mundo.
       ¿Debemos preocuparnos ante la eventualidad de que tengan razón?

`Agradecemos al In.Co.`