Ciencias sociales computacionales

Práctica 5 – Ejercicios de parcial

Parte 1 – Lógica

Ejercicio 1

Sean $x, i : \mathbb{Z}$ y $a : c(\mathbb{Z})$, se tienen los siguientes predicados:

1. $P_1 \equiv \{(\forall i : \mathbb{Z})\,(1 \leq i < |a| \rightarrow a[i] \neq x) \wedge (a[0] = x)\}$
2. $P_2 \equiv \{(\forall i : \mathbb{Z})\,(1 \leq i < |a| \rightarrow a[i] \neq x) \vee (a[0] = x)\}$
3. $P_3 \equiv \{(\forall i, j : \mathbb{Z})\,(0 \leq i < j < |a| \rightarrow a[i] < a[j]) \wedge (a[0] = x)\}$

Indicar la relación de fuerza entre ellos. Justifique su respuesta en **cada** caso.

Ejercicio 2

Sea $a : c(\mathbb{Z})$, se tienen los siguientes predicados:

a) $P_1 \equiv \{(\forall i : \mathbb{Z})\,(0 \leq i < |a| \rightarrow a[i] = 0 \vee a[i] = 1)\}$

b) $P_2 \equiv \left\{ \sum_{i=0}^{|a|-1} \beta(a[i] = 1) + \beta(a[i] = 0) = |a| \right\}$

c) $P_3 \equiv \{(\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow (i \mathsf{\ mod\ } 2 = 0 \wedge a[i] = 0) \vee (i \mathsf{\ mod\ } 2 = 1 \wedge a[i] = 1))\}$

Indicar la relación de fuerza entre ellos. Justifique su respuesta en **cada** caso.

Parte 2 – z básico

Ejercicio 3

Escriba la especificación que describa el comportamiento del siguiente programa:

```
siamofuori <- function(A){
  m = 0
  for(i in seq(length(A))){
    for(j in seq(i+1, length(A))){
      n = abs(A[j]-A[i])
      if(n > m) m = n
     }
  }
  return(m)
}
```

Ejercicio 4

Especificar el problema `maxSubSeqPrimos`, que dada una secuencia de números naturales `a`, devuelva alguna de las subsecuencias de primos consecutivos de mayor longitud. Algunos ejemplos:

- `maxSubSeqPrimos(c(4,1,1)) == c()`, ya que ningún número de la secuencia es primo.
- `maxSubSeqPrimos(c(4,10,1,3)) == c(3)`, ya que la única subsecuencia de primos consecutivos es `[3]`.
- `maxSubSeqPrimos(c(7,7,7)) == c(7,7,7)`.
- `maxSubSeqPrimos(c(2,7,9,5,5,3,11)) == c(5,5,3,11)`.
- `maxSubSeqPrimos(c(5,7,4,5,3))` puede devolver tanto `c(5,7)` como `c(5,3)`.

Ayuda: se recomienda fuertemente utilizar predicados auxiliares.

Ejercicio 5

a) Especificar los siguientes predicados auxiliares:

   I) aux $numero(a : c(\mathbb{Z})) : \mathbb{Z}$, que dada una lista de dígitos enteros del 0 al 9, devuelve su valor como número decimal, interpretando los dígitos de izquierda a derecha. Por ejemplo, $numero($`[1,2,5,0]`$) = 1250$.

   II) aux $digitoIesimo(n, i : \mathbb{Z}) : \mathbb{Z}$, que dado un número entero positivo n y un índice i, devuelve el valor del dígito decimal i-ésimo de n, donde el índice 0 se corresponde al que se encuentra más a la derecha. Por ejemplo, $digitoIesimo(3450, 1) = 5$.

   III) aux $esCapicua(n : \mathbb{Z}) : \mathbb{B}$, que dado un número entero positivo n, devuelve verdadero si sus dígitos decimales son capicúa. Por ejemplo, $esCapicua(120) = False$, pero $esCapicua(12021) = True$.

   Ayuda: Puede suponer que cuenta con un predicado auxiliar numDigitos$(n : \mathbb{Z}) : \mathbb{Z}$ que, dado un número n, indica su número de dígitos decimales significativos (ej. numDigitos$(124) = 3$).

b) Especificar el problema buscarCapicua que, dada una lista de dígitos enteros del 0 al 9 que representan un número natural (donde el primer dígito nunca puede ser un 0), devuelve una lista que representa el próximo capicúa. Por ejemplo:

   - buscarCapicua(`c(1)`) = `c(1)`
   - buscarCapicua(`c(1,2)`) = `c(2,2)`
   - buscarCapicua(`c(1,0,0)`) = `c(1,0,1)`
   - buscarCapicua(`c(1,2,7,3)`) = `c(1,3,3,1)`

   Ayuda: Puede utilizar cualquiera de los predicados auxiliares definidos en el ítem anterior.

Ejercicio 6

Sea la siguiente especificación:

problema Snowden$(L : c(\mathbb{Z})) = res : \mathbb{Z}\{$

$$\text{asegura} : res = \sum_{i=0}^{|L|-1} \beta((\exists j : \mathbb{Z})(0 \leq j < |L| \wedge NSA(L[i]) \wedge NSA(L[j]) \wedge Dist_2(i, j)));$$

$$\text{aux } NSA(x : \mathbb{Z}) = \left(\left(\sum_{i=1}^{x-1} i \cdot \beta(x \bmod i = 0)\right) = x\right);$$

$$\text{aux } Dist_2(x, y : \mathbb{Z}) = (y = x + 2);$$

$\}$

Implemente en R una función que utilice ciclos y cumpla con la especificación anterior.

Puede utilizar funciones auxiliares si lo considera necesario. No se evalúa que la sintaxis sea correcta. Es decir, se espera que la implementación funcione salvo **detalles** de sintaxis.

Ejercicio 7

Sea la siguiente especificación:

problema f4($a : c(\mathsf{Char})) = res : \mathbb{B}${

asegura : $res \leftrightarrow ((\forall c_1, c_2 : \mathsf{Char})(ord(``a") \leq ord(c_1) < ord(c_2) \leq ord(``z") \rightarrow cuenta(a, c_1) < cuenta(a, c_2)) \wedge (\forall c : \mathsf{Char})(ord(``a") \leq ord(c) \leq ord(``z") \rightarrow cuenta(a, c) > 0))$;

aux $cuenta(a : c(\mathbb{T}), c : \mathbb{T}) : \mathbb{Z} = \sum_{i=0}^{|a|-1} \beta\,(a[i] = c)$;

}

Implemente en R una función que utilice ciclos y cumpla con la especificación anterior.

Puede utilizar funciones auxiliares si lo considera necesario. No se evalúa que la sintaxis sea correcta. Es decir, se espera que la implementación funcione salvo **detalles** de sintaxis.

Nota: puede asumir que los caracteres (tipo Char) se encuentran ordenados de la manera esperada.

Parte 3 – Inducción

Ejercicio 8

Sea la siguiente definición:

$$\begin{cases} f(0) & = 2 \\ f(1) & = 3 \\ f(n+1) & = 3 \cdot f(n) - 2 \cdot f(n-1) \quad \text{si } n > 1 \end{cases}$$

Probar que $(\forall n : \mathbb{N})(f(n) = 2^n + 1)$

Ejercicio 9

Sea f tal que $f : \mathbb{N} \rightarrow \mathbb{N}$ y además: $f(1) = 25$ y $\forall n \in \mathbb{N}$, $n \geq 1$, $f(n+1) = f(n) + 4$

a) Dar una fórmula general para f en términos de n.

b) Demostrar por inducción la fórmula encontrada en a).

Ejercicio 10

Sea $(a_n)_{n \in \mathbb{N}}$ la sucesión de números enteros definida recursivamente por

$$a_1 = 1 \qquad a_{n+1} = 2 \cdot a_n + 1, \quad \forall n \in \mathbb{N}$$

Probar que $a_n = 2^n - 1$

Ejercicio 11

Sea $(a_n)_{n \in \mathbb{N}}$ la sucesión de números enteros definida recursivamente por

$$a_1 = 1 \qquad a_2 = 8 \qquad a_n = a_{n-1} + 2 \cdot a_{n-2} \qquad (\forall n \geq 3)$$

Probar que $a_n = 3 \cdot 2^{n-1} + 2 \cdot (-1)^n, \quad \forall n \in \mathbb{N}$

Ayuda: recordar el principio de inducción completa, que dice que sea $p(n)$, $n \in \mathbb{N}$, si se satisfacen $p(1)$ y $p(2)$, y vale el paso inductivo $(\forall h : \mathbb{N})((p(h) \wedge p(h+1)) \rightarrow p(h+2))$, entonces $(\forall n : \mathbb{N})(p(n))$.

Parte 4 – Recursión

Ejercicio 12

Dada la especificación:

problema elPajaroCantaHastaMorir$(X, V : c(\mathbb{Z})) = sal : c(\mathbb{Z})\{$
asegura : $(\forall i : \mathbb{Z})(0 \leq i < |X| \rightarrow ((pert(X[i], sal) \wedge \neg pert(X[i], V)) \vee (\neg pert(X[i], sal) \wedge pert(X[i], V)))\wedge$
$0 \leq |sal| \leq |X|$;
aux $pert(T : c(\mathbb{Z}), a : \mathbb{Z}) = (\exists j : \mathbb{Z})(0 \leq j < |T| \wedge T[j] = a)$;
}

Implemente esta función utilizando *recursión*. No se evalúa que la sintaxis sea correcta. Es decir, se espera que la implementación funcione salvo **detalles** de sintaxis.

Ejercicio 13

Dada la siguiente especificación:

problema suma$(A, B : c(\mathbb{Z})) = sal : c(\mathbb{Z})\{$
requiere : $|B| > 0$;
asegura : $|sal| = |A| \wedge (\forall i : \mathbb{Z})\,(0 \leq i < |A| \rightarrow sal[i] = (A[i] + B[i \,\mathsf{mod}\, |B|]))$;
}

Implemente esta función utilizando *recursión*. Puede utilizar funciones auxiliares si lo considera necesario, pero todas deben ser recursivas. No se evalúa que la sintaxis sea correcta. Es decir, se espera que la implementación funcione salvo **detalles** de sintaxis.

Ejercicio 14

Implementar una función que utilice **recursión** y que cumpla con la especificación que se detalla a continuación. No se evalúa que la sintaxis sea correcta. Es decir, se espera que la implementación funcione salvo **detalles** de sintaxis.

problema listaPrimos$(x, n : \mathbb{Z}) = res : [\mathbb{Z}]\{$
requiere : $n \geq 0$;
asegura : $|res| = n \wedge (\forall i : \mathbb{Z})(0 \leq i < n \rightarrow esPrimo(res[i]) \wedge iesimoDesdeX(x, i, res[i]))$;
aux $iesimoDesdeX(x, i, val : \mathbb{Z}) : \mathbb{B} = \left(\sum_{j=x}^{val} \beta(esPrimo(j))\right) - 1 = i$;
aux $esPrimo(n : \mathbb{Z}) : \mathbb{B} = (n > 1 \wedge \neg(\exists k : \mathbb{Z})(2 \leq k < n \wedge n \mod k = 0))$;
}

Nota: Puede definir funciones auxiliares, siempre y cuando éstas también sean recursivas (o no tengan ciclos). También puede suponer que cuenta con una función booleana `esPrimo(n)` que indica si `n` es un número primo.

Ejercicio 15

Implementar una función que utilice **recursión** y que cumpla con la especificación que se detalla a continuación. No se evalúa que la sintaxis sea correcta. Es decir, se espera que la implementación funcione salvo **detalles** de sintaxis.

problema checkParentesis$(a : c(\text{Char})) = res : \mathbb{B}\{$
asegura : $res \leftrightarrow ((\forall j : \mathbb{Z})(0 \leq j < |a| \rightarrow cuentaHasta(a, j, \text{“(”}) \geq cuentaHasta(a, j, \text{“)”}) \wedge cuentaHasta(a, |a|-1, \text{“(”}) = cuentaHasta(a, |a|-1, \text{“)”})));$

aux $cuentaHasta(a : c(\mathbb{T}), i : \mathbb{Z}, x : \mathbb{T}) : \mathbb{Z} = \sum_{j=0}^{i} \beta(a[j] = x);$

$\}$

Por ejemplo:

1. `checkParentesis(string("(2+5/2+Esteban")) == False`
2. `checkParentesis(string("(Maxi/(((5+5)))*(4+2)/(5+5)+...+3)")) == True`
3. `checkParentesis(string("((5))(2)((1)))(")) == False`
4. `checkParentesis(string("")) == True`

Nota: Puede definir funciones auxiliares, siempre y cuando éstas también sean recursivas (o no tengan ciclos).