Práctica 2 – Python básico

Parte 1 – Problemas algebraicos

Ejercicio 1

Especifique e implemente las siguientes funciones *booleanas*:

a) **noEsCero(n)**: devuelve True si n es distinto de cero

problema noEsCero$(n : \mathbb{R}) = res : Bool\{$
requiere: $True$
asegura:$res == (n \neq 0)$
$\}$

b) **iguales(n1, n2)**: devuelve True si n1 es igual a n2.

problema iguales$(n1, n2 : t) = res : Bool\{$
requiere: $True$
asegura:$res == (n1 == n2))$
$\}$

c) **menor(n1, n2)**: devuelve True si n1 es menor (estricto) a n2.

problema menor$(n1, n2 : t) = res : Bool\{$
requiere: $True$
asegura:$res == (n1 < n2)$
$\}$

d) **par(n)**: devuelve True si n es un número par.

problema par$(n : \mathbb{Z}) = res : Bool\{$
requiere: $True$
asegura:$res == (n \mod 2 == 0)$
$\}$

e) **divisible(n, d)**: devuelve True si n es divisible por d.

problema divisible$(n, d : \mathbb{Z}) = res : Bool\{$
requiere: $d \neq 0$
asegura:$res == (n \mod d == 0)$
$\}$

f) **imparDivisiblePorTresOCinco(n)**: devuelve True si n es divisible por 3 o por 5 pero no por 2.

problema imparDivisiblePorTresOCinco$(n : \mathbb{Z}) = res : Bool\{$
requiere: $True$
asegura:$res == (((n \mod 3 == 0) \vee (n \mod 5 == 0)) \wedge (n \mod 2 \neq 0))$
$\}$

Ejercicio 2

Especifique e implemente las siguientes funciones sobre enteros:

a) **factorial(n)**: devuelve el valor del factorial de n.

problema factorial$(n : \mathbb{Z}) = res : \mathbb{Z}\{$
requiere: $n \geq 0$
asegura:$res == \prod_{i=1}^{n} i$
$\}$

b) **sumaDivisores(n)**: devuelve la suma de todos los divisores positivos de n.

problema sumaDivisores$(n : \mathbb{Z}) = res : \mathbb{Z}\{$
requiere: $True$
asegura:$res == \sum_{i=1}^{|n|} \beta(n \mod i == 0) * i$
$\}$

c) **primo(n)**: devuelve True si n es un número primo.

problema primo$(n : \mathbb{Z}) = res : Bool\{$
requiere: $True$
asegura:$esPrimo(n)$
$\}$

$esPrimo(n : \mathbb{Z}) \equiv \{(n \neq 0 \rightarrow (res == (\forall k : \mathbb{Z})(1 < k < |n| \rightarrow n \mod k \neq 0))) \wedge (n == 0 \rightarrow res == False)\}$

d) **menorDivisiblePorTres(n)**: dado un n positivo, devuelve el menor número mayor a n tal que sea divisible por 3.

problema menorDivisiblePorTres$(n : \mathbb{Z}) = res : \mathbb{Z}\{$
requiere: $n > 0$
asegura:$res \mod 3 == 0 \wedge res > n$
asegura: $(\forall k : \mathbb{Z})(n < k < res \rightarrow k \mod 3 \neq 0)$
$\}$

e) **mayorPrimo(n1, n2)**: devuelve True si n1 es el mayor primo que divide a n2.

problema mayorPrimo$(n1, n2 : \mathbb{Z}) = res : Bool\{$
requiere: $True$
asegura:$res == (esPrimo(n1) \wedge (n2 \mod n1 == 0) \wedge (\forall k : \mathbb{Z})((n1 < k \leq n2 \wedge esPrimo(k)) \rightarrow n2 \mod k \neq 0)$
$\}$

f) **mcd(n1, n2)**: devuelve el máximo común divisor entre n1 y n2.

problema mcd$(n1, n2 : \mathbb{Z}) = res : \mathbb{Z}\{$
requiere: $\neg(n1 == 0 \wedge n2 == 0)$
asegura:$(n1 \mod res == 0) \wedge (n2 \mod res == 0)$
asegura:$(\forall k : \mathbb{Z})(res < k \leq |n2| \rightarrow \neg(n1 \mod k == 0 \wedge n2 \mod k == 0))$ $\}$

Parte 2 – Secuencias

Ejercicio 3

Especifique e implemente las siguientes funciones sobre secuencias. Para la implementación, en los casos en los que exista una función equivalente en Python, no está permitido utilizarla

a) **suma(a)**: devuelve la suma de todos los elementos de la lista a

Siendo $\mathbb{T}$ un tipo que soporte la suma:

problema suma$([a : \mathbb{T}]) = res : \mathbb{T}\{$
requiere: $True$
asegura:$res == suma(a)$
$\}$

$suma([a : \mathbb{T}]) \equiv \{\sum_{i=0}^{|a|-1} a[i]\}$

¿Está bien que si la lista es vacía la suma de 0?

b) **promedio(a)**: devuelve el promedio de todos los elementos de la lista a. ¿Qué ocurre si a no tiene elementos?

Siendo $\mathbb{T}$ un tipo que soporte la suma y la división:

problema promedio$([a : \mathbb{T}]) = res : \mathbb{T}'\{$
requiere: $|a| > 0$
asegura:$res == suma(a)/|a|$
$\}$

Si aceptáramos como entrada válida la lista vacía, el asegura se indefiniría.

c) **maximo(a)**: devuelve el máximo entre todos los elementos de la lista a.

problema maximo$([a : \mathbb{T}]) = res : \mathbb{T}\{$
requiere: $|a| > 0$
asegura:$(\exists i : \mathbb{Z})(0 \leq i < |a| \rightarrow a[i] == res)$
asegura: $esCotaSuperior(res, a)$
$\}$

$esCotaSuperior(val : \mathbb{T}, [a : \mathbb{T}]) \equiv \{(\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow val \geq a[i])\}$

d) **listaDeAbs(a)**: devuelve una lista con los valores absolutos de cada elemento de la lista a.

problema listaDeAbs$([a : \mathbb{T}]) = res : [\mathbb{T}]\{$
requiere: $True$
asegura:$(\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow pertenece(|a[i]|, res))$
asegura:$(\forall x : \mathbb{T})(cantApariciones(x, res) == cantApariciones(x, a) + cantApariciones(-x, a))$
$\}$

$cantApariciones(val : \mathbb{T}, [a : \mathbb{T}]) \equiv \{\sum_{i=0}^{|a|-1} \beta(a[i])\}$

$pertenece(val : \mathbb{T}, [a : \mathbb{T}]) \equiv \{cantApariciones(val, a) \neq 0\}$

e) **todosPares(a)**: devuelve True si todos los elementos de la lista a son pares.

problema todosPares$([a : \mathbb{Z}]) = res : Bool\{$
requiere: $True$
asegura:$res == (\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow a[i] \mod 2 == 0)$
$\}$

f) **maximoAbsoluto(a)**: devuelve el máximo entre los valores absolutos de todos los elementos de la lista a.

problema maximoAbsoluto$([a : \mathbb{T}]) = res : \mathbb{T}\{$
requiere: $|a| > 0$
asegura:$(\exists i : \mathbb{Z})(0 \leq i < |a| \rightarrow |a[i]| == res)$
asegura: $(\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow res \geq |a[i]|)$
$\}$

g) **divisores(n)**: devuelve una lista con todos los divisores positivos de n

problema divisores$([a : \mathbb{Z}]) = res : [\mathbb{Z}]\{$
requiere: $n \neq 0$
asegura:$(\forall k : \mathbb{Z})(1 \leq k \leq |n| \wedge n \mod k == 0 \rightarrow pertenece(k, res))$
$\}$

Obs: res tiene que tener todos los divisores positivos, pero no dice nada sobre elementos repetidos ni sobre contener números que no sean divisores.

Si además quiero que en res no haya ningún número que no sea divisor agrego:

$$(\forall i : \mathbb{Z})(0 \leq i < |res| \rightarrow n \mod res[i] == 0)$$

Si además no quiero que haya elementos repetidos agrego:

$$(\forall i, j : \mathbb{Z})((0 \leq i < |res| \wedge 0 \leq j < |res| \wedge i \neq j) \rightarrow res[i] \neq res[j])$$

h) **cantidadApariciones(a, x)**: devuelve la cantidad de veces que aparece el elemento x en la lista a.

problema cantidadApariciones$([a : \mathbb{T}], x : \mathbb{T}) = res : \mathbb{Z}\{$
requiere: $True$
asegura:$res == cantApariciones(x, a)$
$\}$

i) **masRepetido(a)**: devuelve el elemento que más veces aparece repetido en la lista a.

problema masRepetido$([a : \mathbb{T}]) = res : \mathbb{T}\{$
requiere: $|a| > 0$
asegura:$(\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow cantApariciones(res, a) \geq cantApariciones(a[i], a))$
$\}$

j) **ordenAscendente(a)**: devuelve True si todos los elementos de la lista a aparecen en orden ascendente. Ejemplos:

- ordenAscendente([]) == True
- ordenAscendente([1,2,4]) == True
- ordenAscendente([4,2,1]) == False

problema ordenAscendennte($[a : \mathbb{T}]$) = $res : Bool\{$
requiere: $True$
asegura:$(\forall i : \mathbb{Z})(0 \leq i < |a| - 1 \rightarrow a[i] < a[i+1])$
}

k) **reverso(a)**: devuelve una lista que cumple que sus elementos son los mismos que los de a, pero se encuentran en el orden inverso. Ejemplos:

- reverso(['h','o','l','a']) == ['a','l','o','h']
- reverso(reverso(['h','o','l','a'])) == ['h','o','l','a']

problema reverso($[a : \mathbb{T}]$) = $res : [\mathbb{T}]\{$
requiere: $True$
asegura $|res| == |a|$
asegura:$(\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow res[i] == a[|a| - 1 - i])$
}

Ejercicio 4

Implemente funciones en Python que cumplan con las siguientes especificaciones. Proponga un nombre declarativo en castellano para cada función implementada.

a) **problema A**($n : \mathbb{Z}$) = $x : \mathbb{R}\{$
requiere: $n \geq 0$;
asegura:$x^2 == n$;
}

Nombre propuesto: raizCuadrada

Explicación de la implementación:

$$x = \sqrt{a}$$

Podemos hallar la raiz cuadrada $x \in \mathbb{Z}$ de un numero entero a hallando el $x \in \mathbb{Z}$ más grande tal $x * x \leq a$ mediante un ciclo **while**. El problema es que la especificación nos pide devolver $\mathbb{R}$. Lo más cercano a un $\mathbb{R}$ que podemos devolver es un *float* con cierta cantidad de cifras después de la coma.

Notar que multiplicar un numero por diez es correr la coma un lugar a la derecha, entonces $\sqrt{a} * 10^d$ representa a la $\sqrt{a}$ con la coma corrida d lugares a la derecha.

Si metemos a 10^d dentro de la raiz nos queda $\sqrt{a} * 10^d = \sqrt{a * 10^{2d}}$ entonces

$$\sqrt{a} = \frac{\sqrt{a * 10^{2d}}}{10^d}$$

De este modo al hallar a x', el entero más grande tal que $x' * x' \leq \sqrt{a * 10^{2d}}$ estaríamos hallando $\sqrt{a} * 10^d$. De este modo para obtener $\sqrt{a}$ solo faltaría dividir x' por 10^d.

b) **problema B**($[a : \mathbb{Z}]$) = $x : \mathbb{Z}\{$
requiere: $True$;
asegura:$x == (\sum_{i=0}^{|a|-1} \beta(i \mod 2 == 0) \cdot a[i]$;
}

Nombre propuesto: sumaPosPares

c) **problema C**($[a : \mathbb{Z}]$) = $b : \mathbb{B}\{$
requiere: $True$;
asegura:$b == (\forall i : \mathbb{Z})(0 \leq i < |a| \rightarrow (a[i] == a[|a| - 1 - i]))$;
}

Nombre propuesto: capicua

d) **problema D**$([a : \mathbb{Z}]) = r : \mathbb{Z}\{$
requiere: $|a| > 0$;
asegura:$r == (\sum_{i=0}^{|a|-1} \beta(i \mod 2 == 1) \cdot a[i])/(\frac{|a|}{2})$;
$\}$

Nombre propuesto: promedioImpares

e) **problema E**$([a : \mathbb{Z}]) = r : \mathbb{Z}\{$
requiere: $|a| > 0$;
asegura:$(\exists i : \mathbb{Z})(0 \leq i \wedge i < |a| \wedge (\forall j : \mathbb{Z})(0 \leq j \wedge j < |a| \rightarrow a[i] \leq a[j]) \wedge r == a[i])$;
$\}$

Nombre propuesto: minimo

f) **problema F**$([a : \mathbb{Z}]) = r : \mathbb{Z}\{$
requiere: $|a| > 0$;
asegura:$(\exists i, j : \mathbb{Z})(0 \leq i \wedge i \leq j \wedge j < |a| \wedge todosIgualesEntreIndices(i, j, a) \wedge ((\forall l, m : \mathbb{Z})(0 \leq l \wedge l \leq m \wedge m < |a| \wedge todosIgualesEntreIndices(l, m, a) \rightarrow j - i \geq m - l) \wedge r == j - i)$;
$\}$

$todosIgualesEntreIndices(i, j : \mathbb{Z}, [a : \mathbb{Z}]) \equiv \{(\forall k : \mathbb{Z})(i \leq k \wedge k < j \rightarrow a[k] == a[i])\}$

Nombre propuesto tamañoDeLaMayorSublistaTodosIguales