UBA – Facultad de Ciencias Exactas y Naturales – Departamento de Computación– Algoritmos y Estructura de Datos I

Primer parcial – 17/10/2018

1		2		3			TOTAL
a	b	a	b	a	b	c	
20	0	10	15	5	5	28	83 (A)

hojas: 4

FILO

Aclaraciones: Se permite tener UNA hoja A4 con anotaciones durante el parcial. Cualquier decisión de interpretación que se tome debe ser aclarada y justificada. Para aprobar se requieren al menos 60 puntos.
Entregar cada ejercicio en una hoja separada, numerada y que incluya el nro. de orden.

Ejercicio 1. [35 puntos]

a) **[30 puntos]** Especificar el siguiente problema. Dada una secuencia de enteros s, devuelve res==True si en la lista, todos son números primos, a excepción de uno solo que debe ser la productoria de todos los otros primos. Si la propiedad no se cumple res debe ser False.

Por ejemplo, si la entrada es:

- $s =< 11, 154, 2, 7, >$ entonces res==True
- $s =< 3, 5, 15 >$ entonces res==True
- $s =< 2, 7, >$ entonces res==False
- $s =< 2, 7, 10, >$ entonces res==False (pues 10 no es 7*2)
- $s =< 2, 2, 7, 11, 308 >$ entonces res==True (pues 308=2*2*7*11*308)

```
proc producTodosPrimosSalvoUno (in s: seq⟨ℤ⟩, out res: Bool) {
    Pre {True}
    Post {..}
}
```

Puede usar sin tener que especificar el predicado esPrimo(in x: $\mathbb{Z}$)

b) **[5 puntos]** *Si se se agregara al problema un out resN, que tiene que devolver el valor de la productoria. ¿Qué debería cambiar en la precondición?*

Ejercicio 2. [25 puntos] Dada la siguiente especificación junto con el siguiente programa:

```
proc fibonacciExtendido (in s: seq⟨ℤ⟩, out res: Bool) {
    Pre {S = S0 ∧ |S| ≥ 3}
    Post {S = S0 ∧ res = True ↔ (∀j : ℤ)(2 ≤ j < |S| →L S[j] =
          S[j − 1] + S[j − 2])}
}
```

```
res := True;
i := 2;
while (i < |s|) do
  if (S[i] != S[i-1] + S[i-2]) then
    res := False
  endif;
  i:=i+1
endwhile
```

a) **[10 puntos]**Proponer el invariante del ciclo en palabras.

b) **[15 puntos]**Escribir el invariante con lógica, Pc y Qc

Ejercicio 3. [40 puntos]

Dado el siguiente programa, invariante , Pc y Qc

- $P_c : (res1 == 0) \wedge (res2 == 0) \wedge (i == 0)$
- $Q_c : (i == |s|) \wedge (res1 = \sum_{j=0}^{i-1} IFesPrimo(j) THEN\ s[j]\ ELSE\ 0) \wedge (res2 = \sum_{j=0}^{i-1} IFesPrimo(j) THEN\ 0\ ELSE\ s[j])$
- $I : (0 \leq i \leq |s|) \wedge_L (res1 = \sum_{j=0}^{i-1} IFesPrimo(j) THEN\ s[j]\ ELSE\ 0) \wedge (res2 = \sum_{j=0}^{i-1} IFesPrimo(j)\ THEN\ 0\ ELSE\ s[j])$

```
res1 := 0;
res2 :=0
i := 0;
while (i < |s|) do
  if ( esPrimo(i) ) then
        res1 := res1+s[i]
  else
        res2:= res2+s[i]
  endif;
  i:=i+1
endwhile
```

realizar los siguientes pasos de la demostración:

a) **[5 puntos]** $P_c \implies I$

b) **[5 puntos]** $(I \wedge \neg B) \implies Q_c$

c) **[30 puntos]** $\{I \wedge B\} S \{I\}$

① (a)

proc productoDosPrimosSalvoUno (in s: seq<ℤ>, out res: Bool) {

Pre {True}

Post {res = True ⇔ (∃ j:ℤ)((0 ≤ j < |s|) ∧_L (s[j] = productoDemas(j,s)) ∧

(∀ i:ℤ)((0 ≤ i < j ∨ j < i < |s|) →_L esPrimo(s[i]))) }

aux productoDemas(j:ℤ, s:seq<ℤ>) = $\prod_{i=0}^{j-1} s[i] \cdot \prod_{i=j+1}^{|s|-1} s[i]$;

}

X LO IDEA ESTA BIEN, FALLA [1] = 5

(b) se debería incluir un |s| ≥ 2, ya que si |s| es 1 o 0 la productoria sería de elementos indefinidos ya que no quedarían posiciones para multiplicar y comparar.

X NO SOLO ESO, SINO, QUE PASARIA SI LE 5 NO CUMPLE CON LA PROP, OSEA NO HAY NINGUN VALOR QUE CUMPLA QUE SEA LA PRODUCTORIA DE LOS PRIMOS, INCLUSO QUE PASA SI NO HAY PRIMOS ?

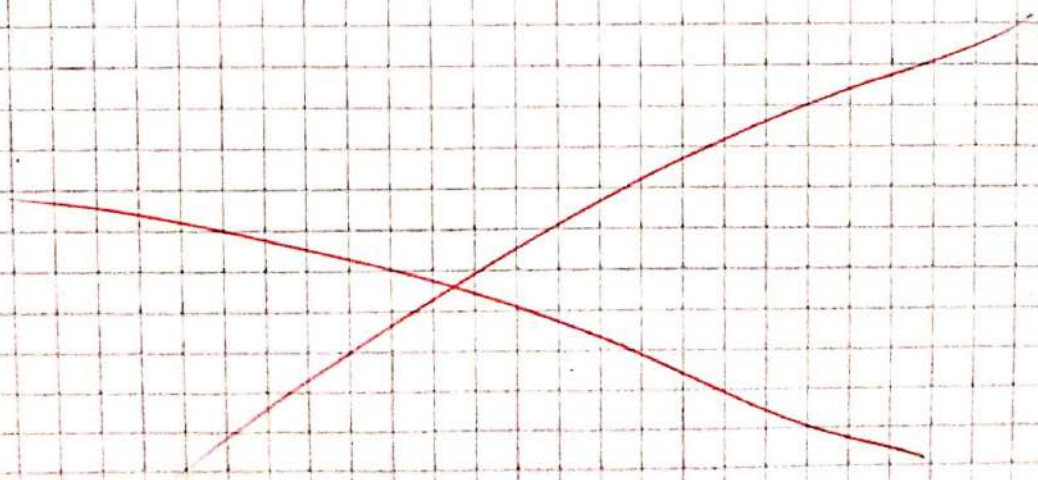
② (a) Antes y después de cada iteración del ciclo se cumple que la secuencia S sobre la que se trabaja es igual a la secuencia dada originalmente, y que para una variable i que itera entre todos los valores desde 2 hasta la longitud de la secuencia S inclusive, res es verdadero sí y sólo sí todos los elementos de la secuencia entre la segunda posición (inclusive) y la posición i-ésima (no inclusive) son iguales a la suma de los dos elementos que los anteceden. ✓

(b) $I \equiv 0 \leq i \leq |S| \wedge S = S_0 \wedge res = True \leftrightarrow (\forall j : \mathbb{Z})(2 \leq j < i \rightarrow_L S[j] = S[j-1] + S[j-2])$

$P_c \equiv res = True \wedge i = 2 \wedge S = S_0 \wedge |S| \geq 3$

$Q_c \equiv S = S_0 \wedge i = |S| \wedge res = True \leftrightarrow (\forall j : \mathbb{Z})(2 \leq j < i \rightarrow_L S[j] = S[j-1] + S[j-2])$

✓



③ (a) $P_c \Rightarrow I$

Hiremos la precondición del ciclo y al invariante en partes separadas por y-lógicos y veamos que P_c implica a I:

- $res1 = 0 \wedge i = 0 \Rightarrow (0 \leq i \leq |s|) \wedge_L (res1 = \sum_{j=0}^{i-1} \text{if} \cdots \text{else } 0)$

$$\equiv (0 \leq 0 \leq |s|) \wedge_L (0 = \sum_{j=0}^{-1} \text{if} \cdots \text{else } 0)$$

$$\equiv \text{True} \wedge_L (0 = 0)$$

$$\equiv \text{True} \wedge_L \text{True} \equiv \text{True}$$

- $res2 = 0 \wedge i = 0 \Rightarrow (0 \leq i \leq |s|) \wedge_L (res2 = \sum_{j=0}^{i-1} \text{if} \cdots \text{else } s[j])$

$$\equiv (0 \leq 0 \leq |s|) \wedge_L (0 = \sum_{j=0}^{-1} \text{if} \cdots \text{else } s[j])$$

$$\equiv \text{True} \wedge_L (0 = 0)$$

$$\equiv \text{True} \wedge_L \text{True} \equiv \text{True}$$

Quedan probadas esas implicaciones, por lo tanto se sigue que $res1 = 0 \wedge res2 = 0 \wedge i = 0 \Rightarrow I$

$P_c \Rightarrow I$ ■.

(b) $(I \wedge \neg B) \Rightarrow Q_c$

$I \wedge \neg B \equiv \underbrace{i = |s|}_{(0 \le i \le |s| \wedge i \ge |s|)} \wedge_L (res1 = \sum_{j=0}^{i-1} \text{if} \dots \text{else } 0) \wedge (res2 = \sum_{j=0}^{i-1} \text{if} \dots \text{else } s[j])$

Podemos ver entonces que $I \wedge \neg B$ y Q_c son exactamente iguales. Como $p \Rightarrow p$ para cualquier fórmula lógica, entonces $(I \wedge \neg B) \Rightarrow Q_c$ ■ ✓

(c) queremos ver que $\{I \wedge B\}\ S\ \{I\}$ es válida, por lo tanto debemos verificar que $I \wedge B \Rightarrow wp(S, I)$

Veamos quién es $wp(S, I)$

$wp(S, I) \equiv wp(\text{if } B \text{ then } S1 \text{ else } S2 ;\ i := i+1,\ I)$

con $B \equiv esPrimo(i)$, $S1 \equiv res1 := res1 + s[i]$, $S2 \equiv res2 := res2 + s[i]$

$\equiv wp(\text{if } B \text{ then } S1 \text{ else } S2, \underbrace{wp(i := i+1, I)}_{Q_1}) \equiv$ (*)

$Q_1 \equiv \underbrace{def(i+1)}_{\equiv True} \wedge_L \underbrace{(0 \le i+1 \le |s|)}_{\equiv -1 \le i < |s|} \wedge_L (res1 = \sum_{j=0}^{i} \text{if} \dots) \wedge (res2 = \sum_{j=0}^{i} \text{if} \dots)$

(*) $\equiv \underbrace{def(esPrimo(i))}_{\equiv True} \wedge_L ((esPrimo(i) \wedge wp(S1, Q_1)) \vee (\neg esPrimo(i) \wedge wp(S2, Q_1))) \equiv$ (**)

mino aparte:

$wp(S1, Q_1) \equiv def(res1 + s[i]) \wedge_L -1 \le i < |s| \wedge_L \underbrace{(res1 + s[i] = \sum_{j=0}^{i} \text{if} \dots)}_{} \wedge$

$(res2 = \sum_{j=0}^{i} \text{if} \dots)$

ESTO PODÍS CONTRADECIRLA ACÁ sab si $s[i]$ es Primo, COSA QUE NO SABÉS PORQUE NO METISTE PARA ADENTRO EL esPrimo(i)

$\equiv 0 \le i < |s| \wedge_L (res1 = \sum_{j=0}^{i-1} \text{if} \dots) \wedge (res2 = \sum_{j=0}^{i} \text{if} \dots)$ ✗

$wp(S2, Q_1) \equiv_{def} (res2 + s[i]) \wedge_L -1 \leq i < |s| \wedge_L$

$(res1 = \sum_{j=0}^{i} if\ldots) \wedge (res2 + s[i] = \sum_{j=0}^{i} if--)$

x MISMO LADO $\longrightarrow$

$\equiv 0 \leq i < |s| \wedge_L (res1 = \sum_{j=0}^{i} if\ldots) \wedge (res2 = \sum_{j=0}^{i-1} if--)$

$\equiv$ [A]

$(esPrimo(i) \wedge 0 \leq i < |s| \wedge_L (res1 = \sum_{j=0}^{i-1} \text{if } esPrimo(j) \text{ then } s[j] \text{ else } 0)$

$\wedge (res2 = \sum_{j=0}^{i} \text{if } esPrimo(j) \text{ then } 0 \text{ else } s[j])) \vee$

[B]

$(\neg esPrimo(i) \wedge 0 \leq i < |s| \wedge_L (res1 = \sum_{j=0}^{i} \text{if } esPrimo(j) \text{ then } s[j] \text{ else } 0)$

$\wedge (res2 = \sum_{j=0}^{i-1} \text{if } esPrimo(j) \text{ then } 0 \text{ else } s[j]))$

miro A

para que esta cadena de y-lógicos tenga valor de verdad true, todos los términos deben ser verdaderos simultáneamente. Tomando entonces que esPrimo(i) es true y $0 \leq i < |s|$ también, miro

$res2 = \sum_{j=0}^{i} \text{if } esPrimo(j) \text{ then } 0 \text{ else } s[j]$

como tomé esPrimo(i) true, es igual a

$res2 = \sum_{j=0}^{i-1} \text{if } esPrimo(j) \text{ then } 0 \text{ else } s[j] + 0$

miro B

de manera análoga, sabiendo que esPrimo(i) es false, (así $\neg esPrimo(i)$ es true), el término

$res1 = \sum_{j=0}^{i} \text{if } esPrimo(j) \text{ then } s[j] \text{ else } 0$

queda

$res1 = \sum_{j=0}^{i-1} \text{if } esPrimo(j) \text{ then } s[j] \text{ else } 0 + 0$

Volviendo a (1), nos queda

$\equiv (esPrimo(i) \wedge C) \vee (\neg esPrimo(i) \wedge C)$

con $C \equiv 0 \leq i < |s| \wedge_L (res1 = \sum_{j=0}^{i-1} if\ldots) \wedge (res2 = \sum_{j=0}^{i-1} if\ldots)$

por lógica, esto es igual a

$\equiv C \wedge (esprimo(i) \vee \neg esPrimo(i))$

$\equiv C$

por ende, $wp(S,I) \equiv C$

Ahora nos falta ver que $I \wedge B \Rightarrow \underbrace{wp(S,I)}_{\equiv C}$

$I \wedge B \equiv 0 \leq i < |s| \wedge_L (res1 = \sum_{j=0}^{i-1} \text{if } esPrimo(j) \text{ then } s[j] \text{ else } 0)$

$\wedge (res2 = \sum_{j=0}^{i-1} \text{if } esPrimo(j) \text{ then } 0 \text{ else } s[j])$

Pero podemos ver que $I \wedge B$ y C son iguales!

Por lo tanto, por lógica se sigue que

$I \wedge B \Rightarrow C$ ■ ✓