ISIS 1105 Diseño y Análisis de Algoritmos **Nombre: ______________________________**
Semestre 2014-10 – Parcial 1 **Sección:1**
Febrero 25, 2014
Prof. Rodrigo Cardoso

1 *[20/100]*

Enriquezca GCL con una instrucción FOR, de la forma

`for (INIC ; B ; INCR) S rof`

con semántica operacional descrita así:

i. Se ejecuta la instrucción INIC.
ii. Si el predicado B no vale, se termina la ejecución. En otro caso se continúa.
iii. Se ejecuta la instrucción S.
iv. Se ejecuta la instrucción INCR.
v. Se va al paso ii.

1a *[10/20]* Simule **for** (INIC ; B ; INCR) S **rof** con un programa GCL que no utilice esta instrucción.

```
FOR   ≅    INIC;
           do B  →    S;
                      INCR
           od
```

[10/10]

1b *[10/20]* Enuncie un "Teorema del FOR" que permita concluir la corrección de afirmaciones como

{Q} **for** (INIC ; B ; INCR) S **rof** {R}

Como hay un **do** … **od** involucrado, es necesario considerar un invariante P y una cota t adecuadas. También debe adicionarse una aserción intermedia Q1, dentro del cuerpo de la iteración. La anotación adicional de aserciones sería de la forma:

```
FOR   ≅    INIC;

           {Inv P}
           {cota t}
           do B  →    S;
                      {Q1}
                      INCR
           od
```

Con esta notación se puede enunciar el siguiente teorema:

Teo (Corrección de FOR):

Si se cumple que:

(1) P vale antes : {Q} INIC {P]
(2) P sirve : $P \wedge \neg B \Rightarrow R$
(3) P invariante : [a] $\{P \wedge B\}$ S {Q1}
[b] {Q1} INCR {P}

(4) Hay terminación

Entonces:

$$\{Q\}\ \mathtt{FOR}\ \{R\}$$

[10/10]

Variante (terminación formal)

Debe tenerse una función cota `t` de la que se pueda afirmar (para formalizar la condición (4) del teorema del FOR):

(4a) $P \wedge B \Rightarrow t>0$
(4b) $\{P \wedge B \wedge t = t_0\}\ S\ \{t < t_0\}$

[10/10]

2 ***[40/100]***
El siguiente algoritmo sirve para decidir si un patrón `x[0..m-1]` aparece en una cadena `s[0..n-1]`, donde `1≤m≤n`.

```
[Ctx: x[0..m-1]: int  ∧  s[0..n-1]: int ∧  1≤m<n
{Pre Q: true}
{Pos R: (0≤a≤n-m ∧ (∀i| 0≤i<m : x[i]=s[a+i])) ∨ (a>n-m ∧ "no hay match")}

 ⌈ a:= 0;
   match:= false;

   {Inv P1}
   {cota t1}
   do    a≤n-m  ∧ ¬match
         →    a:= a+1;
              r:= 0;
              match:= true;

              {Inv P2}
              {cota t2}
              do r≠m ∧ match
                    →  match:= match ∧ (x[r] = S[a+r]);
                        r:= r+1
              od
   od
⌋
```

2a ***[20/40]*** Enuncie todas las *obligaciones de prueba*, i.e., las condiciones que deben cumplirse para garantizar la corrección del algoritmo, con las anotaciones dadas. *No es necesario incluir las condiciones para demostrar la terminación.*

[1] P1 vale antes : `{Q} a:= 0;  match:= false {P1}`

[2] P1 sirve : `P1 ∧ (a>n-m ∨ match) ⇒ R`

[3] P1 invariante :

```
{P1 ∧ a≤n-m  ∧ ¬match}
a:= a+1;
r:= 0;
match:= true;
```

```
{Inv P2}
{cota t2}
do r≠m ∧ match
      →    match:= match ∧ (x[r] = S[a+r]);
           r:= r+1
od
{P1}
```

Para mostrar la condición [3]:

[3.1] P2 vale antes:

```
{P1 ∧ a≤n-m  ∧ ¬match}
a:= a+1;
r:= 0;
match:= true;
{P2}
```

[3.2] P2 sirve :

```
P2 ∧ (r=m ∨ match)  ⇒ P1
```

[3.3] P2 invariante:

```
{P2 ∧ r≠m ∧ match}
match:= match ∧ (x[r] = S[a+r]);
r:= r+1
{P2}
```

[20/20]

2b ***[20/40]*** Encuentre una función f que estime T(n), la complejidad temporal del algoritmo en el peor caso, como $\theta(f(n))$. Considere como operaciones básicas las comparaciones de elementos de los arreglos.

AYUDAS:
(1) Estime la complejidad en el peor caso como una expresión que dependa de m y de n.
(2) Note que m depende de n. Considere fija la n y encuentre el máximo buscado.

El ciclo externo del algoritmo se hace, en el peor caso, $\theta(n-m)$ veces, puesto que a empieza en 0 y puede terminar en n-m+1.

El ciclo interno se hace, en el peor caso, $\theta(m)$ veces, porque puede pasar que el patrón se revise siempre hasta el final.

Por tanto:

$$T(n) = \theta(n-m)*\theta(m)$$
$$= \theta(m(n-m))$$

[10/20]

Ahora, nótese que T(n) se está expresado en términos de una función $g(m)= m(n-m)$, con $1\leq m\leq n$. Para determinar el peor caso, debe encontrarse el valor máximo de la función g.

Como g es una función continua y derivable en m, sus valores extremos (máximos o mínimos) están en los extremos del intervalo o en un valor donde la derivada sea 0, lo que corresponde a un extremo local.

En este caso:

$g'(m) = n-2m.$

Hay un extremo local en n=m/2.

$g''(m) = n > 0$

Por tanto, el extremo en m=n/2 es un máximo local: $g(n/2) = n^2/4$ es un máximo local.

Los valores de g en los extremos del intervalo son

$g(1) = n-1$
$g(n) = 0$

y hay que ver cuándo son mayores que $n^2/4$. Es fácil mostrar que, para n>2, el valor máximo se alcanza en n/2. Por tanto.

$$\begin{aligned} T(n) &= \theta(g(n/2)) \\ &= \theta(n^2/4) \\ &= \theta(n^2) \end{aligned}$$

[10/20]

3 ***[30/100]***

La población de un cultivo de bacterias se denota p(n), n≥1. El día 1 hay un solo individuo y se ha determinado que, al final del día n, n>1, hay tantas bacterias como la suma de la población en cada uno de los días anteriores, aumentada con el cuadrado de n.

3a ***[25/30]*** Encuentre un expresión cerrada para p(n), n≥1.

La población obedece a las siguientes condiciones de definición:

$p(1) = 1$
$p(n) = (+i|\ 1\le i<n : p(i)) + n^2, \qquad n\ge 2.$

[5/25]

Nótese que, para n≥1:

$p(n+1)$
$=$
$(+i|\ 1\le i<n+1 : p(i)) + (n+1)^2$
$=$
$(+i|\ 1\le i<n : p(i)) + p(n) + (n+1)^2$
$=$
$p(n) - n^2 + p(n) + (n+1)^2$
$=$
$2*p(n) + 2*n + 1$

Entonces:

$p(n+1) - 2*p(n) = 2*n + 1 \quad , n\ge 1$

[5/25]

$\equiv$
$(E-2)p = \langle 2*n+1\rangle$
$\Rightarrow \quad \langle (E-1)^2$ anula a $\langle n\rangle$ y a $\langle 1\rangle\ \rangle$
$(E-1)^2(E-2)p = 0$

[5/25]

Deben existir constantes A, B, C tales que

$$p(n) = A + B*n + C*2^n \quad , n\ge 1$$

[5/25]

Ahora:

$p(1) = 1$
$p(2) = a(1) + 2^2 = 1 + 4 = 5$
$p(3) = a(1) + a(2) + 3^2 = 1 + 5 + 9 = 15$

Por tanto;

$1 = A + B + 2*C$
$5 = A + 2*B + 4*C$
$15 = A + 3*B + 8*C$

Resolviendo:
A = -3
B = -2
C = 3

i.e.,

$$p(n) = 3*2^n - 2*n - 3 \quad , n \geq 1$$

[5/25]

3b ***[5/30]*** Encuentre una función f.n tal que $p(n) = \theta(f(n))$.

```
   p(n)
=      ⟨3a⟩
   3*2^n – 2*n – 3
=      ⟨propiedades de θ⟩
   θ(2^n)
```

[5/5]