ISIS 1105 Diseño y Análisis de Algoritmos **Nombre:** ______________________________
Semestre 2015-10 – Parcial 1 **Sección: 1**
Febrero 24, 2015
Prof. Rodrigo Cardoso

1 ***[30/100]***

Si para dos programas, S1 y S2, se define S1 $\sqsupseteq$ S2 (S1 *simula* S2) así:

$$S1 \sqsupseteq S2 \equiv (\forall Q,R|: \{Q\}\ S2\ \{R\} \Rightarrow \{Q\}\ S1\ \{R\})$$

y se define que S1 $\cong$ S2 (S *equivale a* S2), de modo que

$$S1 \cong S2 \equiv (S1 \sqsupseteq S2) \wedge (S2 \sqsupseteq S1)$$

1a *[15/30]* Pruebe que

$$S1 \cong S2 \equiv (\forall Q,R|: \{Q\}\ S2\ \{R\} \equiv \{Q\}\ S1\ \{R\})$$

$(\forall Q,R\ |: \{Q\}\ S2\ \{R\} \equiv \{Q\}\ S1\ \{R\})$
$=$ $\langle$Def $\equiv\rangle$
$(\forall Q,R\ |: (\{Q\}\ S2\ \{R\} \Rightarrow \{Q\}\ S1\ \{R\}) \wedge (\{Q\}\ S1\ \{R\} \Rightarrow \{Q\}\ S2\ \{R\}))$
$=$ $\langle$Distr $\forall/\wedge\rangle$
$(\forall Q,R\ |: (\{Q\}\ S2\ \{R\} \Rightarrow \{Q\}\ S1\ \{R\})) \wedge (\forall Q,R\ |: (\{Q\}\ S1\ \{R\} \Rightarrow \{Q\}\ S2\ \{R\}))$
$=$ $\langle$Def $\sqsupseteq\rangle$
$(S1 \sqsupseteq S2) \wedge (S2 \sqsupseteq S1)$
$=$ $\langle$Def $\cong\rangle$
$S1 \cong S2$

[15/15]

Variante 1

($\Rightarrow$) Supóngase que S1 $\cong$ S2. Entonces S1 $\sqsupseteq$ S2 y S2 $\sqsupseteq$ S1.
Sean Q, R predicados.
Si $\{Q\}S2\{R\}$, deberá cumplirse que $\{Q\}S1\{R\}$, porque S1 $\sqsupseteq$ S2.
También, si $\{Q\}S1\{R\}$, deberá cumplirse que $\{Q\}S2\{R\}$, porque S2 $\sqsupseteq$ S1.
Es decir, $\{Q\}S2\{R\}$ equivale a $\{Q\}S1\{R\}$.
Como Q, R son arbitrarios, se tiene que
$S1 \cong S2 \Rightarrow (\forall Q,R|: \{Q\}\ S2\ \{R\} \equiv \{Q\}\ S1\ \{R\})$

($\Leftarrow$) Supóngase que $(\forall Q,R|: \{Q\}\ S2\ \{R\} \equiv \{Q\}\ S1\ \{R\})$. Entonces, para Q, R arbitrarios:
$\{Q\}\ S2\ \{R\} \Rightarrow \{Q\}\ S1\ \{R\}$ y $\{Q\}\ S1\ \{R\} \Rightarrow \{Q\}\ S2\ \{R\}$.
Es decir:
$(\forall Q,R\ |: (\{Q\}\ S2\ \{R\} \Rightarrow \{Q\}\ S1\ \{R\}))$ y $(\forall Q,R\ |: (\{Q\}\ S1\ \{R\} \Rightarrow \{Q\}\ S2\ \{R\}))$
o bien:
$(S1 \sqsupseteq S2)$ y $(S2 \sqsupseteq S1)$
Por tanto: S1 $\cong$ S2.

[15/15]

Variante 2

$S1 \sqsupseteq S2$
$=$

$(\forall Q,R|: \{Q\}\ S2\ \{R\} \Rightarrow \{Q\}\ S1\ \{R\})$
$\Rightarrow$
$\{wp(S2|R)\}\ S2\ \{R\} \Rightarrow \{wp(S2|R)\}\ S1\ \{R\}$
$=$
$\{wp(S2|R)\}\ S1\ \{R\}$
$=$
$wp(S2|R) \Rightarrow wp(S1|R)$

Es decir: $S1 \sqsupseteq S2 \Rightarrow (wp(S2|.) \Rightarrow wp(S1|.))$

Si $(wp(S2|.) \Rightarrow wp(S2|.))$:
$\{Q\}\ S2\ \{R\}$
$=$
$Q \Rightarrow wp(S2|R)$
$\Rightarrow$
$Q \Rightarrow wp(S1|R)$
$=$
$\{Q\}\ S1\ \{R\}$

O sea: $(wp(S2|.) \Rightarrow wp(S1|.)) \Rightarrow S1 \sqsupseteq S2$

Por tanto: $S1 \sqsupseteq S2 \equiv (wp(S2|.) \Rightarrow wp(S1|.))$

Y además: $S1 \cong S2 \equiv wp(S2|.) \equiv wp(S1|.)$

Ahora:
$S1 \cong S2$
$=$
$wp(S2|.) \equiv wp(S1|.)$
$\Rightarrow$
$wp(S2|R) \equiv wp(S1|R)$
$\Rightarrow$
$(Q \Rightarrow wp(S2|R)) \equiv (Q \Rightarrow wp(S1|R))$
$=$
$\{Q\}S2\{R\} \equiv \{Q\}S1\{R\}$.

Y, si esto es cierto para todo Q,R ¸se muestra que
$(\forall Q,R|: \{Q\}\ S2\ \{R\} \equiv \{Q\}\ S1\ \{R\}) \Rightarrow S1 \cong S2$

[15/15]

1b *[15/30]* Considere el programa:

$\mathbf{if}\ B \rightarrow S\ \square\ \neg B \rightarrow S\ \mathbf{fi}$

Pruebe que $\mathbf{if}\ B \rightarrow S\ \square\ \neg B \rightarrow S\ \mathbf{fi} \cong S$.

Sean Q, R predicados arbitrarios. Entonces:
$\{Q\}\ \mathbf{if}\ B \rightarrow S\ \square\ \neg B \rightarrow S\ \mathbf{fi}\ \{R\}$
$=$ $\langle$Regla Hoare - IF$\rangle$
$(Q \Rightarrow B \vee \neg B) \wedge \{Q \wedge B\}\ S\ \{R\} \wedge \{Q \wedge \neg B\}\ S\ \{R\}$
$=$ $\langle B \vee \neg B \equiv true;\ Q \Rightarrow true;\ true \wedge \alpha \equiv \alpha \rangle$
$\{Q \wedge B\}\ S\ \{R\} \wedge \{Q \wedge \neg B\}\ S\ \{R\}$
$=$ $\langle \{Q1\}\ S\ \{R\} \wedge \{Q2\}\ S\ \{R\} \equiv \{Q1 \vee Q2\}\ S\ \{R\} \rangle$
$\{(Q \wedge B) \vee (Q \wedge \neg B)\}\ S\ \{R\}$

$=\qquad\langle \text{Distr } \wedge/\vee\rangle$

$\{Q \wedge (B \vee \neg B)\}\ S\ \{R\}$

$=\qquad\langle B \vee \neg B \equiv \text{true};\ \text{true} \wedge \alpha \equiv \alpha\rangle$

$\{Q\}\ S\ \{R\}$

Ahora, por **1a** (y generalización):

$\textbf{if } B \to S \ \square\ \neg B \to S \textbf{ fi} \ \cong\ S$

[15/15]

2 ***[30/100]***

Sea $a(n)$ una sucesión tal que:

$a(0) = 1$

$a(1) = 2$

$a(n) = a(n-1)$, si $n\geq 2$, n par

$ = a(n-2) + 1$, si $n\geq 2$, n impar

2a *[20/30]* Defínanse dos secuencias basadas en a:

$b(n) = a(2n)$, $n\geq 0$

$c(n) = a(2n+1)$, $n\geq 0$

Encuentre expresiones cerradas para $b(n)$ y para $c(n)$.

AYUDA: solucione primero $c(n)$.

Solución para c:

$c(0) = a(1)$

$ = 2$

$c(n+1) = a(2n+3)$, $n\geq 0$

$ = a(2n+1) + 1$, $n\geq 0$

$ = c(n) + 1$, $n\geq 0$

Es decir:

$(E-1)c = \langle 1\rangle$

$\Rightarrow$

$(E-1)^2 c = 0$

Por tanto, existen constantes A, B tales que:

$c(n) = A + Bn$, $n\geq 0$

Resolviendo:

$c(n) = n+2$, $n\geq 0$

[10/20]

Solución para b:

$b(0) = a(0)$

$ = 1$

$b(n) = a(2n)$, $n\geq 1$

$ = a(2n-1)$, $n\geq 1$

$ = c(n-1)$, $n\geq 1$

$ = n+1$, $n\geq 1$

[10/20]

2b *[10/30]* Encuentre una función f tal que $a = \theta(f)$

Nótese que:

```
a(n)      = b(n/2)          , n≥0, n par
          = c((n-1)/2)      , n≥0, n impar
```

Por tanto:

```
a(n)      = (n+1)/2         , n≥0, n par
          = (n-1)/2 + 2     , n≥0, n impar
```

Es decir:

```
a(n) = θ(n).
```

[10/10]

3 *[40/100]*

La función `sel` sirve para calcular el `i`-simo elemento en un arreglo de enteros `a[p..q-1]`. Utiliza como subrutina la función `part`.

Dado un arreglo b, `perm(b)` representa un arreglo con los mismos valores de b, permutados.

```
function sel (a[p..q-1]:int,i:nat): int
{Pre QS: a = A ∧ p≤i<q }
{Pos RS: a = perm(A) ∧  a[p..i-1] ≤ a[i] < a[i+1..q-1]  ∧ sel = a[i]}
  ⌈ if  p=q-1   →    sel:= a[p];

    [] p≠q-1    →    r:= part(a[p..q-1],a[q-1]);
                     k:= r-p+1;
                     if    i=k   →    sel:= a[r]
                     []    i<k   →    sel(a[p..k-1],i)
                     []    i>k   →    sel(a[r+1..q-1],i-k)
                     fi
   fi;

   sel:= a[i]

   ⌋

function part (a[p..q-1]:int, x:int): nat
{Pre QP: a = A }
{Pos RP: a = perm(A) ∧  a[p..i] ≤ x < a[i+1..q-1] ∧ part = i}

  ⌈ (1)   i,j:= p,q;

          {Inv P: a = perm(A) ∧  a[p..i] ≤ x < a[j+1..q-1] }
          {cota t: j-i}
    (2)   do i≠j   →   (2.1) if a[i]≤x  →    (2.1a)      i:= i+1
                             [] a[i]>x  →    (2.1b.1)    j:= j-1;
                                             (2.1b.2)    a[i],a[j]:= a[j],a[i]
                             fi
          od;

          {Q1: a = perm(A) ∧  a[p..i] ≤ x < a[i+1..q-1]}
    (3)   part:= i
  ⌋
```

3a ***[10/40]*** Enuncie todas las *obligaciones de prueba*, i.e., las condiciones que deben cumplirse para garantizar la corrección del algoritmo de la función `part`, con las anotaciones dadas. Use las etiquetas para no reescribir el código, pero no deje indicadas instrucciones que no sean secuencias de asignaciones. *No es necesario incluir las condiciones para demostrar la terminación.*

[1] P vale antes: $\{QP\}$ (1) $\{P\}$

[2/10]

[2] P sirve : $P \wedge i=j \Rightarrow Q1$

[2/10]

[3] P invariante : $\{P \wedge i \neq j\}$ (2.1) $\{P\}$

[3.1] : $P \wedge i \neq j \Rightarrow (a[i] \leq x \vee a[i] > x)$

[3.2] : $\{P \wedge i \neq j \wedge a[i] \leq x\}$ i:= i+1 $\{P\}$

[3.3] : $\{P \wedge i \neq j \wedge a[i] > x\}$ j:= j-1; a[i],a[j]:= a[j],a[i] $\{P\}$

[4/10]

[4] R vale al final: $\{Q1\}$ (3) $\{R\}$

[2/10]

3b ***[15/40]*** Llamando n a q-p (la dimensión del arreglo para partir), encuentre una función f que estime $T_{part}(n)$, la complejidad temporal de `part` en el peor caso, como $\theta(f(n))$. Considere como operación básica la asignación.

$T_{part}(n) = T(1) + T(2) + T(3)$

T(1) y T(3) son asignaciones, cada una con costo O(1).

T(2) es un ciclo que se repite q-1-p+1=n veces. Para ver esto, nótese que cada elemento del arreglo se analiza exactamente una vez en el ciclo. O bien: la función cota empieza en q-p=n y disminuye en 1 en cada iteración.

Resumiendo:

$$
\begin{aligned}
T_{part}(n) &= \theta(1) + \theta(n) + \theta(1) \\
&= \theta(n).
\end{aligned}
$$

[15/15]

3c ***[15/40]*** Llamando n a q-p (la dimensión del arreglo para encontrar el i-simo elemento), encuentre una función g que estime $T_{sel}(n)$, la complejidad temporal de `sel` en el peor caso, como $\theta(g(n))$. Considere como operación básica la asignación.

La recurrencia de `sel` es sobre un arreglo de tamaño k-p o sobre un arreglo de tamaño q-1-k, con p≤k<q. En las dos alternativas el peor caso se da sobre un arreglo de tamaño q-p-1 = n-1.

Entonces:

$$
\begin{aligned}
T_{sel}(n) &= 2 && \text{, si } n=1 \\
&= T_{part}(n) + 1 + \max(1, T_{sel}(n-1)) && \text{, si } n>1 \\
&= \theta(n) + T_{sel}(n-1) && \text{, si } n>1
\end{aligned}
$$

Entonces, se puede remplazar $\theta(n)$ por γn (γ constante no nula) y afirmar que:

$T_{sel}(n+1) = \gamma n + T_{sel}(n)$, si $n>1$

=

$(E-1)T_{sel} = \langle \gamma n \rangle$

⇒

$(E-1)^3 T_{sel} = 0$

Deben existir A, B, C constantes, tales que:

$T_{sel}(n) = A + Bn + Cn^2$, $n \geq 0$

Resolviendo:

$T_{sel}(n) = 2 - \gamma n/2 + \gamma n^2/2$, $n \geq 0$

De modo que:

$T_{sel}(n) = \theta(n^2)$

[15/15]