



**Integrantes:** Andrés M. Hense, Victoria Espil

**Ejercicio 14.a** Especificar los siguientes problemas:

- Dado un número entero positivo, obtener la suma de sus factores primos.

**Respuesta**

```
proc sumaFactoresPrimos (in n:  $\mathbb{Z}$  ,out result:  $\mathbb{Z}$ ) {  
  Pre { $n > 0$ }  
  Post {  
    result =  $\sum_{i=2}^n$  if esPrimo( $i$ ) then  $i$  else 0 fi  
  }  
}
```

Notar que si  $n < 2$  entonces la sumatoria devolvera 0, por definición de como se comportan las sumatorias.

**Ejercicio 15.f** Especificar los siguientes problemas sobre secuencias:

- Dadas dos secuencias  $s$  y  $t$ , devolver su *intersección*, es decir, una secuencia con todos los elementos que aparecen en ambas. Si un mismo elemento tiene repetidos, la secuencia retornada debe contener la cantidad mínima de apariciones entre  $s$  y  $t$ .

**Respuesta**

```
proc interseccion (in l : seq<T> ,in m : seq<T> ,out res : seq<T>) {  
  Pre {True}  
  Post {  
    perteneceAAmbos(l, m, res)  
     $\wedge_L$   
    mismaCantRep(l, m, res)  
  }  
}  
  
pred perteneceAAmbos(l, m, res : seq<T>) {  
  res  $\in$  l  $\wedge$  res  $\in$  m  
}  
  
pred mismaCantRep(l, m, res : seq<T>) {  
  ( $\forall j : \mathbb{Z}$ )( $0 \leq j < |res|$ )  $\rightarrow_L$  cantRep(res, res[j]) = minRep(l, m, res[j])  
}  
  
aux minRep(l, m : seq<T> ,n : T) :  $\mathbb{Z}$  {  
  if cantRep(l, n) < cantRep(m, n) then cantRep(l, n) else cantRep(m, n) fi  
}  
  
aux cantRep(l: seq<T> , n : T) :  $\mathbb{Z}$  {  
   $\sum_{i=0}^{|l|-1}$  if l[i] = n then 1 else 0 fi  
}
```

*Dudas:* Tiene que tener algo la **Pre?**, ese *perteneceAAmbos*, tranquilamente puede volar, o así como esta, esta demasiado escueto?, como me asegura esta implementación que *res* no tiene elementos de más?.

**Ejercicio 22.a** Especificar los siguientes problemas de modificación de secuencias:

- **proc primosHermanos**(inout  $l : seq\langle\mathbb{Z}\rangle$ ), que dada una secuencia de enteros mayores a dos, reemplaza dichos valores por el número primo menor más cercano. Por ejemplo, si  $l = \langle 6, 5, 9, 14 \rangle$ , luego de aplicar **primosHermanos**( $l$ ),  $l = \langle 5, 5, 7, 13 \rangle$

## Respuesta

```

proc primosHermanos(inout  $l : seq\langle\mathbb{Z}\rangle$ ) {
  Pre {  $l = l_0 \wedge$ 
     $(\forall number : \mathbb{Z})(number \in l \rightarrow number > 2)$  }
  Post {
     $|l_0| = |l|$ 
     $\wedge_L$ 
     $(\forall i : \mathbb{Z})(0 \leq i < |l_0|) \rightarrow_L l_0[i] = menorPrimoCerc(l[i])$ 
  }
}

```

```

aux menorPrimoCerc( $n : \mathbb{Z}$ ) :  $\mathbb{Z}$  {
  if esPrimo( $n$ ) then n else menorPrimoCerc( $n - 1$ ) fi
  juajua, re que no se puede usar recursion en un aux }

```

*Dudas:* lo ideal seria poder hacer un `proc menorPrimoCerc` que use un `while`, pero no creo que se pueda hacer eso, y así como esta no le veo la vuelta.