Paradigmas de la Programación
Primer Parcial

Gabriel Infante-Lopez Franco M. Luque Walter Alini

30 de abril de 2009

1. Considere el siguiente código:

```
fun {FoldFactory F U}
   fun {FoldR L F U}
      case L
      of nil then U  <------ (3)
      [] X|L2 then
         {F X  {FoldR L2 F U}}
      end
   end
in
   fun {$ L} {FoldR L F U} end <---- (2)
end
```

```
fun {Sum A B}
   A + B
end
```

```
local G H in
   G = {FoldFactory Sum 0}
   H = {G [1]} <---(1)
   {Browse H}
end
```

   a) Traduzca a lenguaje de kernel.

   b) Ejecute en la máquina abstracta el código que obtuvo del punto anterior. Los estados que nos interesan son los estados a los que la máquina llega cuando las sentencias marcadas con (1), (2) y (3) están como elementos actuales en la pila semántica. Para estos estados reporte claramente:

      1) Entorno Contextual.
      2) Variables y valores en el ASA.
      3) Variables que pueden ser recolectadas por el recolector de basura.

2. La siguiente función {Elegir Xs} toma una lista de listas Xs como argumento:

```
fun {Elegir Xs}
   case Xs
   of nil then nil
   [] X|Xr then
      Y={Elegir Xr}
   in
      if {Length X}>{Length Y} then X else Y end
   end
end
```

   a) ¿Cuál es el resultado de {Elegir [[a]]}?

   b) ¿Cuál es el resultado de {Elegir [[b a] [a] [b c]]}?

   c) Dé una función recursiva a la cola {ElegirConAcc Xs Y} equivalente a {Elegir Xs} que utilice a Y como acumulador.