Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Escuela de Ciencias Exactas y Naturales
Departamento de Ciencia de la Computación
Análisis de Lenguajes de Programación

Nombre y Apellido:

Examen Parcial 2

1. A continuación se presenta un λ-cálculo tipado extendido con naturales y listas de naturales.

Sintaxis:

$$
\begin{array}{rl}
t ::= & x \mid \lambda x : T.\ t \mid t\ t \mid 0 \mid \texttt{suc}\ t \\
& \mid \texttt{nil} \mid \texttt{cons}\ t\ t \mid \texttt{head}\ t \mid \texttt{R}\ t\ t\ t \\
v & ::= \lambda x : T.\ t \mid nv \mid lv \\
nv & ::= 0 \mid \texttt{suc}\ nv \\
lv & ::= \texttt{nil} \mid \texttt{cons}\ nv\ lv \\
T & ::= T \to T \mid \mathbb{N} \mid \mathbb{N}^*
\end{array}
$$

Sistema de tipos:

$$\Gamma \vdash 0 : \mathbb{N} \qquad \text{(T-Zero)}$$

$$\frac{\Gamma \vdash t : \mathbb{N}}{\Gamma \vdash \texttt{suc}\ t : \mathbb{N}} \qquad \text{(T-Suc)}$$

$$\frac{x : T \in \Gamma}{\Gamma \vdash x : T} \qquad \text{(T-Var)}$$

$$\frac{\Gamma, x : T \vdash t : T'}{\Gamma \vdash \lambda x : T.\ t : T \to T'} \qquad \text{(T-Abs)}$$

$$\frac{\Gamma \vdash t_1 : T \to T' \qquad \Gamma \vdash t_2 : T}{\Gamma \vdash t_1\ t_2 : T'} \qquad \text{(T-App)}$$

$$\Gamma \vdash \texttt{nil} : \mathbb{N}^* \qquad \text{(T-Nil)}$$

$$\frac{\Gamma \vdash t_1 : \mathbb{N} \quad \Gamma \vdash t_2 : \mathbb{N}^*}{\Gamma \vdash \texttt{cons}\ t_1\ t_2 : \mathbb{N}^*} \qquad \text{(T-Cons)}$$

$$\frac{t : \mathbb{N}^*}{\texttt{head}\ t : \mathbb{N}} \qquad \text{(T-Head)}$$

$$\frac{t_1 : T \quad t_2 : \mathbb{N} \to \mathbb{N}^* \to T \to T \quad t_3 : \mathbb{N}^*}{\texttt{R}\ t_1\ t_2\ t_3 : T} \qquad \text{(T-R)}$$

Semántica:

$$\frac{t_1 \to t_1'}{t_1\ t_2 \to t_1'\ t_2} \qquad \text{(E-App1)}$$

$$\frac{t_2 \to t_2'}{v\ t_2 \to v\ t_2'} \qquad \text{(E-App2)}$$

$$(\lambda x : T.\ t_1)\ v \to t_1\ [v\ /\ x] \qquad \text{(E-Abs)}$$

$$\frac{t_1 \to t_2}{\texttt{cons}\ t_1\ t \to \texttt{cons}\ t_2\ t} \qquad \text{(E-Cons1)}$$

$$\frac{t_1 \to t_2}{\texttt{cons}\ v\ t_1 \to \texttt{cons}\ v\ t_2} \qquad \text{(E-Cons2)}$$

$$\frac{t_1 \to t_2}{\texttt{suc}\ t_1 \to \texttt{suc}\ t_2} \qquad \text{(E-Suc)}$$

$$\frac{t \to t'}{\texttt{head}\ t \to \texttt{head}\ t'} \qquad \text{(E-Head1)}$$

$$\texttt{head}\ (\texttt{cons}\ v\ t) \to v \qquad \text{(E-Head2)}$$

$$\frac{t_3 \to t_3'}{\texttt{R}\ t_1\ t_2\ t_3 \to \texttt{R}\ t_1\ t_2\ t_3'} \qquad \text{(E-R1)}$$

$$\texttt{R}\ t_1\ t_2\ \texttt{nil} \to t_1 \qquad \text{(E-R2)}$$

$$\texttt{R}\ t_1\ t_2\ (\texttt{cons}\ v\ t) \to t_2\ v\ t\ (\texttt{R}\ t_1\ t_2\ t) \qquad \text{(E-R3)}$$

a) Enunciar las propiedades de progreso y preservación.

b) Determinar si el lenguaje dado cumple las propiedades de progreso y preservación dando la prueba correspondiente si la propiedad se cumple o dando un contraejemplo en caso contrario.

c) Definir en el lenguaje las siguientes funciones:

   i) La función *tail* que devuelve la cola de una lista.

   ii) La función *map* que aplica una función a todos los elementos de una lista.

d) Se desea extender el lenguaje con un término *iterate* $n\ x$ que construya una lista de n elementos x.
   Extender la gramática, la semántica y el sistema de tipos para contemplar esta nueva construcción.

2. Una forma de representar enteros en λ-calculus consiste en usar un par formado por dos numerales de Church, los cuales representan un valor positivo y un valor negativo, de manera que el valor entero es la diferencia de estos dos números. Por ejemplo, el entero $-x$ podría representarse como $(0, x)$ o como $(y, x + y)$, siendo y cualquier natural.
Usando la representación de pares, booleanos y numerales de Church dadas en clase definir las siguientes funciones en λ-cálculo (suponiendo definidas las funciones *pair*, *fst*, *snd*, *ifthenelse*, *true*, *false*, *succ*, *zero*, *isZero* y *pred*):

1. La función *isPos* determina si un entero es positivo o negativo.
2. La función *neg* que represente la operación negación unaria de un entero.
3. La función *abs* que calcula el valor absoluto de un entero.

3. Dada la representación de naturales y listas en Sistema F:

$$Nat = \forall X.\ (X \to X) \to X \to X$$
$$List\ X = \forall Y.\ (X \to Y \to Y) \to Y \to Y$$

Definir la función $sum : List\ Nat \to Nat$, que suma los elementos de una lista de naturales. Suponer que se tiene definida la función $suma : Nat \to Nat \to Nat$, que suma dos naturales.