9 Semantics of Programming Languages (pes20)

Consider the following pure functional language, in which n ranges over the mathematical integers.

$$
\begin{array}{ll}
T ::= & \mathbf{int}1 \mid \mathbf{int}8 \mid \mathbf{int}16 \mid \mathbf{uint}1 \mid \mathbf{uint}8 \mid \mathbf{uint}16 \mid T \to T' \mid T * T' \mid T + T' \\
e ::= & n \mid e +_T e' \mid x \mid \mathbf{fn}\ x{:}T \Rightarrow e \mid e\ e' \mid (e, e') \mid \#1\ e \mid \#2\ e \mid \mathbf{inl}_T\ e \mid \mathbf{inr}_T\ e \\
 & \mid \mathbf{case}\ e\ \mathbf{of}\ \mathbf{inl}\,(x_1 : T_1) \Rightarrow e_1 \mid \mathbf{inr}\,(x_2 : T_2) \Rightarrow e_2
\end{array}
$$

Its operational semantics is defined as a relation $e \longrightarrow e'$ with the standard rules for a pure call-by-value left-to-right functional language, except with the following rules for addition of values. As usual, the expression $n +_T n'$ is stuck if one of these does not apply.

$$
\frac{\begin{array}{l} n \in -2^{N-1} \ldots 2^{N-1} - 1 \\ n' \in -2^{N-1} \ldots 2^{N-1} - 1 \\ n''{=}n + n' \\ n'' \in -2^{N-1} \ldots 2^{N-1} - 1 \end{array}}{n +_{\mathbf{int}N} n' \longrightarrow n''}\ \text{PLUS\_INT}
\qquad
\frac{\begin{array}{l} n \in 0 \ldots 2^{N} - 1 \\ n' \in 0 \ldots 2^{N} - 1 \\ n''{=}n + n' \\ n''' = n'' \bmod 2^N \end{array}}{n +_{\mathbf{uint}N} n' \longrightarrow n'''}\ \text{PLUS\_UINT}
$$

(*a*) Define a subtype relation $T <: T'$ and type relation $\Gamma \vdash e : T$ for this syntax and operational semantics that will permit flexible use of integers in the appropriate ranges. You can omit the standard type relation rules for the expressions (e, e'), $\#1\ e$, $\#2\ e$, $\mathbf{inl}_T\ e$, $\mathbf{inr}_T\ e$, and **case**. [14 marks]

(*b*) Explain three main aspects of your definitions, with reference to the programming idioms they permit and the runtime errors they exclude, with examples. [6 marks]