Consider the language $L$ below, with call-by-value functions, ML-style references, and types $\text{nat}_+$ and $\text{real}_+$ of positive natural and positive real numbers. $L$ includes a primitive test for primality, $\text{prime}(e)$, and a square-root function, $\text{sqrt}(e)$; these are defined only for positive-natural and positive-real values respectively.

$$T ::= \text{bool} | \text{nat}_+ | \text{real}_+ | T \to T' | T\text{ ref}$$

$$e ::= x | n | r | \text{fn}\ x : T \Rightarrow e | e \ e' | \text{ref}\ e | !e | e ::= e' | \text{prime}(e) | \text{sqrt}(e)$$

Here $x$ ranges over a set $X$ of variables and $n$ and $r$ range over $\mathbb{N}_{>0}$ and $\mathbb{R}_{>0}$ respectively. Let $\Gamma$ range over finite partial functions from $X$ to types $T$.

(a) Give typing rules defining $\Gamma \vdash e : T$ for $\text{prime}(e)$ and $\text{sqrt}(e)$. [1 mark]

(b) There is an obvious runtime coercion from elements of $\text{nat}_+$ to elements of $\text{real}_+$. To let programmers exploit that conveniently, we would like to define a type system for $L$ that includes a subtype relation $T_1 <: T_2$ with $\text{nat}_+ <: \text{real}_+$. The type system should prevent all run-time errors.

(i) Give the other rules defining $T_1 <: T_2$ and the subsumption rule to use that relation in $\Gamma \vdash e : T$. [4 marks]

(ii) Give the 6 (standard) typing rules defining $\Gamma \vdash e : T$ for functions and references. [3 marks]

(iii) With reference to your subtype rule for function types, explain covariance and contravariance of subtyping. Give examples in $L$ showing that your rule is the only reasonable choice. [2 marks]

(iv) Similarly, justify your rule for reference types. [2 marks]

(c) To implement $L$, we want to translate it during typechecking to another typed language $L'$ which makes that coercion explicit where required, as a new expression form $\text{real}\_\text{of}\_\text{nat}(e)$, and which does not have subtyping.

(i) Give the $L'$ typing rule for $\text{real}\_\text{of}\_\text{nat}(e)$ and indicate any other changes required to your type rules for $L$. [1 mark]

(ii) Define an inductive relation $T <: T' \leadsto e$ which for any $T <: T'$ constructs a coercion $e : T \to T'$. [4 marks]

(iii) Define an inductive relation $\Gamma \vdash e \leadsto e'$ where $e$ is an $L$ expression and $e'$ is an $L'$ expression which is like $e$ but with coercions introduced where needed, such that $\Gamma \vdash e : T$ iff $\exists e'$. $\Gamma \vdash e \leadsto e' : T$. You should explain but need not prove that, and you can omit the rules for references. [3 marks]