1 What is curry-howard correspondence?

.
In programming language theory and proof theory, the Curry-Howard correspondence (also known as the Curry-Howard isomorphism or equivalence, or the proofs-as-programs and propositions- or formulae-as-types interpretation) is the direct relationship between computer programs and mathematical proofs.

1.1 type constructions in FP

But before getting into that let's see type constructions in functional programming *Here I'll use SML as an example*

- Tuple type `(int,string)`
  Creating : `val x : int*string = (4 , ''Hello'')`
  Using : `val y : int = #1 x`

- Function type `int -> string`
  Creating : `fun f (x:int) :string = ''the value of x is'' ^ toString(x)`
  Using : `val y: string = f 45`

- Disjunction type `datatype X = Left of int | Right of string`
  Creating : `val x:X = Left 4`
  Creating : `val y:X = Right ''Hello''`
  Using :
  ```
  fun f (x:X) :bool = case x of
                      Left y = ( y > 0 )
                      Right _ = false
  ```
  the above given is a function that takes an argument of type X and returns a value of type bool

- Unit type `type t = unit`
  Creating : `val x:unit = ()`

1.2 From types to propositions

But how does the types defined just translate themselves into propositions? consider the sml code `val x:t = .....` now what does this mean? If this code manages to compile without error then there the type t exists in the system. Let's denote this proposition by $CH(t)$ which means that code has a value of type t or the type t exists in the system.
But that was just for a type t what about all the other things that we have discussed above?

Type	Proposition	Short Notation
t	CH(t)	t
(a,b)	CH(a) and CH(b)	A $\wedge$ B ; A $\times$ B
left a | right b	CH(a) or CH(b)	A $\vee$ B ; A + B
a $\rightarrow$ b	CH(a) implies CH(b)	A $\Longrightarrow$ B
unit	true	1

- type parameter `'a` can be considered as $\forall a$
- consider the function `fun f (x: 'a) : 'a * 'a = (x,x)` this is logically equivalent to $\forall a \quad a \Longrightarrow a \wedge a$