The following is an attempt to write a denotational semantics for Nu's scripting system, based on https://www.youtube.com/watch?v=bmKYiUOEo2A. This presentation uses a more improvised style of syntax than the Conal's, however.

Axiomatic Denotations – When a μ is defined in terms of itself, we consider it axiomatic and irreducible directly in this context.

```
rec μ:Value<a> =
| μ:Value<μ:Relation>
| μ:Value<μ:Address>
| μ:Value<μ:Name>
| μ:Value<μ:String>
| μ:Value<μ:Bool>
| μ:Value<μ:Unit>

and μ:Stream<a> =
| μ:Address -> μ:Stream<a>
| μ:Name -> μ:Relation -> μ:Stream<a>
| μ:Stream<α> -> (α -> a) -> μ:Stream<a>
| μ:Stream<α> -> μ:Stream<β> -> μ:Stream<a when a = α * β)
| μ:Stream<α> -> μ:Stream<β> -> μ:Stream<a when a = α | β)

μ:Effect = μ:Effect // transforms the environment

μ:Declare a = μ:Name -> μ:Declaration a // augments the environment
```

Derived Denotations

```
μ:Get<a> = μ:Name -> μ:Relation -> μ:Value<a>

μ:Set<a> = μ:Name -> μ:Relation -> μ:Value<a> -> μ:Effect

μ:Command<a> = μ:Value<a> -> μ:Effect

μ:Fold<a b> = (μ:Value<a> -> b) -> μ:Stream<a> -> b

μ:Define<a> = μ:Declare<μ:Value<a>>

μ:Variable<a> = μ:Declare<μ:Stream<a>>

μ:Equate<a> = \name -> \rel -> μ:Stream<a> -> μ:Fold (μ:Set<a> name rel)

μ:Handle<a> = μ:Stream<a> -> μ:Fold μ: Command <a>
```