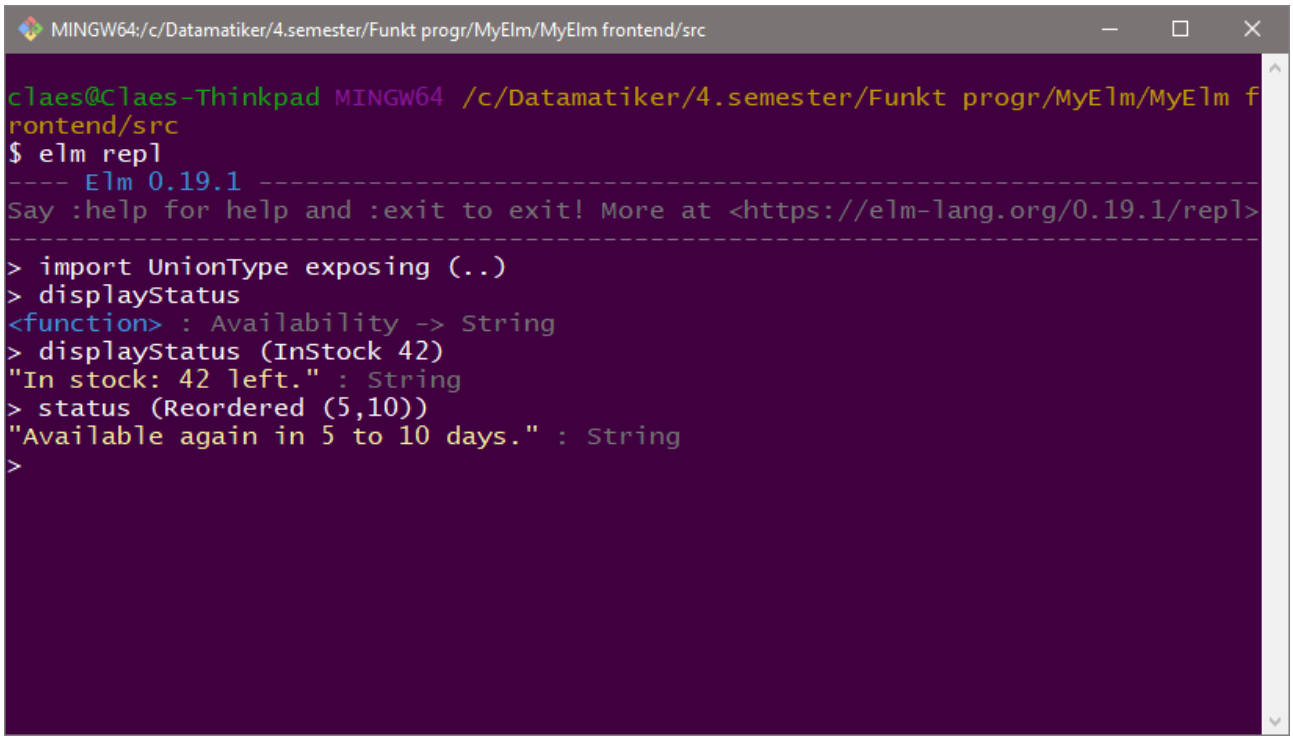
Pixi 4

Custom types (tidligere union types):

Se fil MyElm frontend/CustomType.elm



Haskell:

Type constructor:

```
data Maybe t = Nothing | Just t
```

Maybe t is a type constructor with a type variable, t, while Nothing and Just t are a tagged union where Just t uses the type variable from the type constructor.only one of the three data values can exist at any given point. The symbol **|** expresses the idea of a logical **or** operator. The following code shows how to create and pattern match on the **TrafficLight** type:

```
msgTrafficLight colour =
 case colour of
   Red    -> "you should stop"
```

Yellow -> **"rush before it turns red"**
Green -> **"you can pass"**

The example below redefines **TrafficLight** with additional information, where two data values use the same type, a **String**, and the remaining data value uses the type constructor **Options**.

```
data Options = Rush | SlowDown
data TrafficLight = Red "Stop" | Green "Drive" | Yellow Options
```

```
data Maybe t = Nothing | Just t
```

Maybe t is a type constructor with a type variable, **t**, while **Nothing** and **Just t** are a tagged union where **Just t** uses the type variable from the type constructor.

Error handling example Haskell:

```
head :: [a] -> a
head (x:_) = x
head []    = error "empty list"
```

```
case listToMaybe ages of
  Nothing -> defaultAge
  Just first -> first
```

Elm:

```
> String.toFloat
<function> : String -> Maybe Float

> String.toFloat "3.1415"
Just 3.1415 : Maybe Float

> String.toFloat "abc"
```

```
Nothing : Maybe Float
```

```
type alias User =
  { name : String
  , age : Maybe Int
  }
```

```
sue : User
sue =
  { name = "Sue", age = Nothing }
```

```
tom : User
tom =
  { name = "Tom", age = Just 24 }
```

```
canBuyAlcohol : User -> Bool
canBuyAlcohol user =
  case user.age of
    Nothing ->
      False

    Just age ->
      age >= 21
```

```
Error reporting:
isReasonableAge : String -> Result String Int
isReasonableAge input =
  case String.toInt input of
```

```
Nothing ->
  Err "That is not a number!"

Just age ->
  if age < 0 then
    Err "Please try again after you are born."

  else if age > 135 then
    Err "Are you some kind of turtle?"

  else
    Ok age
```

Level of type constraint in languages: