Exercise 2 for "Dynamic, but Lexical"

Wiktor Kuchta

We are to consider extensions to handlers for a language with lexically-scoped handlers implemented with capabilities.

$$h ::= \texttt{effect}\ x, r \Rightarrow e$$
$$|\texttt{effect}\ x, r \Rightarrow e|^{\ell} = \lambda x.\, \mathsf{shift}_0^{\ell}\, r.\, e$$

We can borrow ideas for restricted forms of operations from Koka. There, the above general form is marked with `ctl`.

Values

$$h ::= ... \mid \texttt{val}\ v$$
$$|\texttt{val}\ v|^{\ell} = v$$

This is useful for providing implicit values without the overhead of delimited control.

```
handle ‘ask in enquirer () with val (fn () => 42)
```

Tail-resumptive operations

$$h ::= ... \mid \texttt{fun}\ x \Rightarrow e$$
$$|\texttt{fun}\ x \Rightarrow e|^{\ell} = \lambda x.\, \mathsf{shift}_0^{\ell}\, r.\, r\, e$$

This common pattern may deserve its own syntax and can be understood as a function executed in the handler's context. It also has the potential for a more efficient implementation.

```
handle ‘ask in enquirer () with fun () => pick (0,1)
```