ひとり Advent Calendar 2016 年 11 月 14 日

これは、ひとり Advent Calendar 2016 7 日目の記事です。

前回は、以下の 2 つに取り組む必要があると書きましたが、その続き。

1. インスタンスのメソッドをちゃんと多相として扱う
2.「それ以外の多相関数（★未実装）」の扱い

まず、前者については、勘違いしていました。IO 向けに特殊化された (specialized) >> は、もう特殊化されているので、もはや多相関数ではありません。なので、これを単相扱いにしていたこと自体は正しいのでした。つまり、これはたまたま動いていたのではなく、意図的にこのように実装したのを、すっかり忘れていただけでした。

ただ、ここでも微妙なところはあって、`p >> q = p >>= \_ -> q` というように、>> のなかで>>= を呼んでいるのですが、これを IO に特殊化された >>= だと決めつけているところは、難があるというか、まちがっていると思う。多相関数の specialization は、やるなら、もっときちんとやるべき。

ま、とりあえずは、このままでいきます。

というわけで、後者。辞書を受け取って、それを、他の多相関数に渡すような多相関数の扱いをやりました。

まず、こういった関数が、いくつの辞書を、どの順で受け取るのかについては、その関数の Qual Type における [Pred] を見ればわかります。

[Pred] が空の場合は、単相関数なので、これまで通りの処理で OK。

[Pred] が空でない場合は、この [Pred] の要素数と同じだけの dictionary 変数を引数に取るので、これを受け取るような Lambda 式で全体をくるんでやって、この Lambda の仮引数を適切に使うように dictionary-passing style 変換してやればよかった。（いまの私自身にしか通じなさそう[*1]な説明ですが）

あ、そうだ、↑これで動いてはいるんだけど、辞書を束縛する変数に TermVar を流用しているのは、いまいちかも。ToDo に書いておこう。

ともあれ、以下のサンプルが動くようになりました。

```
f x y = x > y

s True = "True"
s False = "False"

main = do
  putStrLn $ s (f 100 10)
  putStrLn $ s (f 'x' 'y')
```

実行結果：

[*1] 自分でも、しばらくしてから読んだらわからなさそう。

```
$ ./run test8d/Sample.java
True
False
```

あとは、tcExpr が網羅的でないのを、ちょびちょび補充していけば、完成に近づくんですかね。

それはそうと、DictPass というファイル名、あんまりイケてないなぁと思っていたんだけど、タコっぽくてかわいいような気が、今日急にしてきました。