Assignment 9

Problem 1 and Problem 2 are written in Problem1and2.hs

Problem 3: Type Classes

a.

```
-- Integer comparison
dCompInt :: CompD Int
dCompInt = MakeCompD compareInt
-- List comparison
dCompList :: CompD a -> CompD [a]
dCompList d = MakeCompD compList where
   compList []    []    = Equal
   compList (x:xs) []    = Greater
   compList []    (y:ys) = Less
   compList (x:xs) (y:ys) =
       if ((?=) d x y) /= Equal
      then ((?=) d x y)
      else ((?=) (dCompList d) xs ys)

-- Pair Comparison
dCompPair :: CompD a -> CompD b -> CompD (a, b)
dCompPair da db = MakeCompD compPair where
   compPair (x1, y1) (x2, y2) =
       if ((?=) da x1 x2) /= Equal
      then ((?=) da x1 x2)
      else ((?=) db y1 y2)
```

b.

```
(?=) (dCompPair dCompInt (dCompList dCompChar)) (length "Hello", "Hello") (length "World", "World")
(?=) dCompInt (length “Hello”) (length “World”)
```

```
(?=) (dCompList dCompChar) "Hello"          "World"

(?=) dCompChar          'H'              'W'
```

c.

Type of f: Comp a->[a]->[a]->Comparison

Problem4

(a)-(c) in Problem4.hs

(d) in palindrome.hs

Problem 5

Wrote in Problem5.hs