```haskell
--Aufgabe 1
--a)
class (Eq a) => Rotatable a where
    cwRotCenter, ccwRotCenter, cwRotOrig, ccwRotOrig :: a -> a
    ccwRotCenter = cwRotCenter . cwRotCenter . cwRotCenter
    ccwRotOrig = cwRotOrig . cwRotOrig . cwRotOrig

--b)
data Point = P Float Float deriving (Eq, Ord)
data Rectangle = R Point Float Float deriving Eq

instance Show Point where
    show (P x y) = "Punkt" ++ "(" ++ show x ++ "," ++ show y ++ ")"

instance Show Rectangle where
    show (R (P x y) b h) = "Rechteck" ++ "[" ++ show x ++ "," ++ show (x+b) ++ "]" ++ "x" ++ "[" ++ show y
    ++ "," ++ show (y+h) ++ "]"

--c)
instance Ord Rectangle where
   (R (P x y) b h) <= (R (P x2 y2) b2 h2) = x == x2 && (x+b) == (x2+b2) && y == y2 && (y+h) <= (y2+h2)

--d)
instance Rotatable Point where
    cwRotCenter (P x y) = (P x y)
    cwRotOrig (P x y) = (P y (-x))

instance Rotatable Rectangle where
    cwRotCenter (R (P x y) b h) = (R (P (x+(b-h)/2) (y-(b-h)/2)) h b )
    cwRotOrig (R (P x y) b h) = (R (P y (-x)) h b)

```