可換的な引き算

令和 4 年 6 月 23 日

目錄

1 Implementation of Subtraction **1**

2 Implementation of Addition **2**

2.1 Non-commutative Addition 2

2.2 Commutative Addition 2

```
module Op where
```

1 Implementation of Subtraction

```
data R = R Double deriving Show
```

Defining subtraction as following, hence satisfying $a - b = b - a$.

```
instance Num R where
    (-) (R a) (R b) = case a>=b of
                True -> R $ a-b
                False -> R $ b-a
    fromInteger a = R $ abs x
                    where x = fromIntegral a::Double
    signum (R a)
            | a==0 = 0
            | otherwise = 1
    abs a = a
    (*) _ _ = notImplemented
```

2 Implementation of Addition

2.1 Non-commutative Addition

Now let's try to implement (+) normally

```
(+) (R a) (R b) = R $ a+b
```

But such implementation yields a wierd behaviour where (let a>b) then
$b + (a - b) = a$
$a + (a - b) \neq b$
Which makes the operation non-commutative.

2.2 Commutative Addition

So Let's first find a way to

Extended Definition of Real Number Space

```
data R1 = R1 R Bool deriving Show
data R2 = R2 R Double deriving Show
```

```
instance Num R1 where
    (-) a b = notImplemented
```

```
    abs (R1 a _) = R1 a False
    fromInteger a
                | a>=0 = R1 (R x) False
                | a <0 = R1 (R x) True
                    where x = fromIntegral a::Double
    signum (R1 (R a) d)
                | a==0 = 0
                | d==True = -1
                | d==False = 1
    (+) a b = notImplemented
    (*) _ _ = notImplemented
```