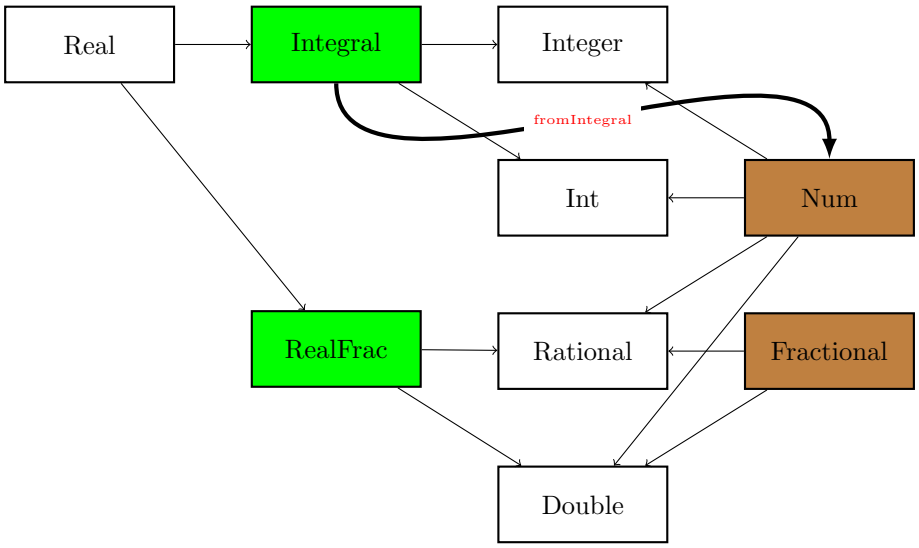


```
-- Ring
class Num a where
    (+):: a -> a -> a
    (/):: a -> a -> a
    (*):: a -> a -> a
    abs:: a -> a
    signum:: a -> a
    negate:: a -> a

-- Field
class (Num a)=> Fractional a where
    (/):: a -> a -> a

class (Num a, Ord a)=> Real a where

-- Integral Domain, Euclidean Domain
-- div, mod, gcd, lcm, etc
class (Real a, Enum a)=> Integral where
```

1 fromIntegral

Any type has **Integral** can be converted to **Num**

Example 1. *Convert o **Int32** Convert **Int32** to **Int***

```
    -- convert Integral to Num
    fromIntegral::(Integral a, Num b)=>a->b
```

2 Converting from and between integral-types(integer-like types)

- Integer - which are arbitrary-precision integer
- Int - which fixed-width machine-specific integers, its range of Int is -2^{31} to $+2^{31} - 1$

3 Converting from and between fractional-types

```
    -- convert from Rational to Fractional
    fromRational::(Fractional a)=>Rational->a
    -- convert from Real to Rational
    toRational::(Real a)=>a->Rational
    -- use Int with '/' division
    let n1 = 3::Int
    let n2 = 4::Int
    n1 / n2 -- error
    fromIntegral n1 / fromIntegral n2
```