

Basic Types



```
1 {-# LANGUAGE FlexibleContexts #-}
2 {-# LANGUAGE MultiParamTypeClasses #-}
3 {-# LANGUAGE OverloadedStrings #-}
4 {-# LANGUAGE QuasiQuotes #-}
5 module Handler.Home where
6
7 import Control.Arrow
8 import Control.Monad
9 import Data.Default
10 import Data.Function
11 import Data.List
12 import Data.Ord
13 import Data.Time
14 import Data.Time.Calendar.WeekDate
15 import Import
16 import Network.HTTP.Types
17 import Yesod.Auth
18
19 getHomeR :: Handler Html
20 getHomeR = undefined
21
22 postHomeR :: Handler Html
23 postHomeR = do
24   ((result, formWidget), formEnctype, formPost) <- runFormPost
25   case result of
26     FormSuccess userRule -> do
27       maybeUserId <- maybeAuthId
28       case maybeUserId of
29         Nothing -> do
30           setErrorMessage [shamlet|You have to be logged in to do that!|]
31           redirect % AuthR.LoginR
32         Just userId -> do
```

Basic Types

Booleans: Bool

Integers: Int, Integer

Reals: Float, Double

Characters: Char

Booleans

Type: `Bool`

Literals: `False` i `True`

Operations:

```
not  :: Bool -> Bool      -- negation
(||)  :: Bool -> Bool -> Bool  -- disjunction
(&&)  :: Bool -> Bool -> Bool  -- conjunction
```

Examples:

```
not True      ↩ False
not False     ↩ True

True || False ↩ True
True && False ↩ False

(False || True) and True ↩ True
not (not True)           ↩ True
not not True             ✗ -- it means: (not not) True
```

Integers

Type:

- `Int`: Integers of 64 bits
- `Integer`: Integers (arbitrarily long)

Literals: `15`, `(-22)`, `857326354873452644468`

Operations: `+`, `-`, `*`, `div`, `mod`, `rem`, `^`.

Relational operators: `<`, `>`, `<=`, `>=`, `==`, `/=` (Δ no `!=`)

Examples:

Integers

Examples:

| | |
|--------------------------|--------------------|
| <code>3 + 4 * 5</code> | <code>23</code> |
| <code>(3 + 4) * 5</code> | <code>35</code> |
| <code>(3 + 4) * 5</code> | <code>35</code> |
| <code>2^10</code> | <code>1024</code> |
| <code>3 + 1 /= 4</code> | <code>False</code> |
| <code>div 11 2</code> | <code>5</code> |
| <code>mod 11 2</code> | <code>1</code> |
| <code>rem 11 2</code> | <code>1</code> |
| <code>mod (-11) 2</code> | <code>1</code> |
| <code>rem (-11) 2</code> | <code>-1</code> |

Reals

Type:

- `Float`: 32-bit floating-point reals
- `Double`: 64-bit floating-point reals

Literals: `3.14`, `1e-9`, `-3.0`

Operations: `+`, `-`, `*`, `/`, `**`.

Relational operators: `<`, `>`, `<=`, `>=`, `==`, `/=`

Integer to Real conversion: `fromIntegral`

Real to Integer conversion: `round`, `floor`, `ceiling`

Reals

Examples:

```
λ> round 3.6
```

```
4
```

```
λ> round (-3.6)
```

```
-4
```

```
λ> map round [3.5, 4.5, 5.5, 6.5]
```

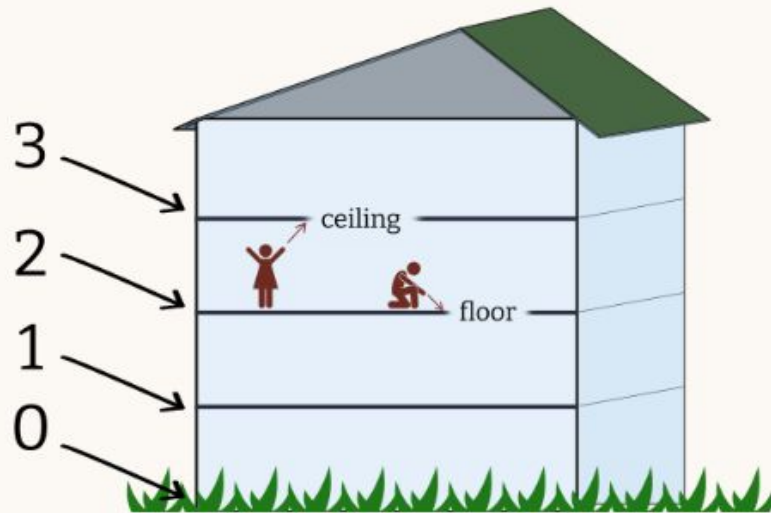
```
[4,4,6,6]
```

Reals

Examples:

```
λ> map ceiling [2.1, 2.2, 2.6, 2.9]  
[3,3,3,3]
```

```
λ> map floor [2.1, 2.2, 2.6, 2.9]  
[2,2,2,2]
```



Reals

Examples:

```
10.0 / 3.0      ➞ 3.3333333333333335
2.0 ** 3.0      ➞ 8.0
fromIntegral 4  ➞ 4.0
```