When you are done you should see the following behavior:

Write a function

A function call is

Function Calls

D.

B.

A.

What is the type of

For example

Multiple Argument Functions

Define

For example

A function that

In Haskell, a

Key Idea:

Recognize Pattern as

Repeated Expressions

Abstraction via Pattern Recognition

Equality-Substitution

Computation via Substituting Equals by Equals

What is Haskell?

From the Lambda Calculus to Haskell

>>>

sumTo n

>>>

20

myMax

>>>

False

>>>

e1 e2

>>>

 quiz x y

quiz ::

pat ::

pat ::

isPos n

isPos ::

isPos ::

A

quiz

quiz ::

ex8

ex8 ::

ex6 ::

ex6 ::

ex5 ::

ex4 ::

ex1

ex1 ::

x_2 ::

$
Let's write a function to

Suppose we have the following

functions consuming lists

How does

We can

As last element is not

First two elements are not

That we can use

We can

A recipe

Syntactic sugar

So we can just avoid the parentheses.

For example

There are two ways to construct lists

When you are done you should see the following behavior

PRACTICE: Clone

Step 3: Write the code

Step 1: Write some tests

A.

B.

C.

D.

EXERCISE

So far: how to

Functions consuming lists

3. Code

j

How does

We can

... but how to

We can

A type for packing

How to return multiple outputs?

 glued xs

5.2

```haskell
sumList :: Int
sumList :: Int
firstElem :: Int
firstElem :: Int
```

```haskell
copy3 x
```
Let's write a function to `take` first `n` elements of a list `xs`:

1. Tests
   ```
   -- >>> ???
   ```
2. Type
   ```
   take :: ???
   ```

**Some useful library functions**

```haskell
-- | Length of the list
length :: [t] -> Int

-- | Append two lists
(++) :: [t] -> [t] -> [t]

-- | Are two lists equal?
(==) :: [t] -> [t] -> Bool
```

You can search for library functions on [Hoogle](http://hoogle.web.haskell.org)! 

**3. Code**

```haskell
'\''haskell
take = ???
```

**Some useful library functions**

```haskell
-- | Length of the list
length :: [t] -> Int

-- | Append two lists
(++) :: [t] -> [t] -> [t]

-- | Are two lists equal?
(==) :: [t] -> [t] -> Bool
```

You can search for library functions on [Hoogle](http://hoogle.web.haskell.org)!

**Recap**

- Core program element is an **expression**
- Every valid expression has a **type** (determined at compile-time)
- Every valid expression reduces to a **value** (computed at run-time)

**Execution**

- Basic values & operators
- Execution / Function Calls just substitute equals by equals
- Pack data into tuples & lists
- Unpack data via **pattern-matching**