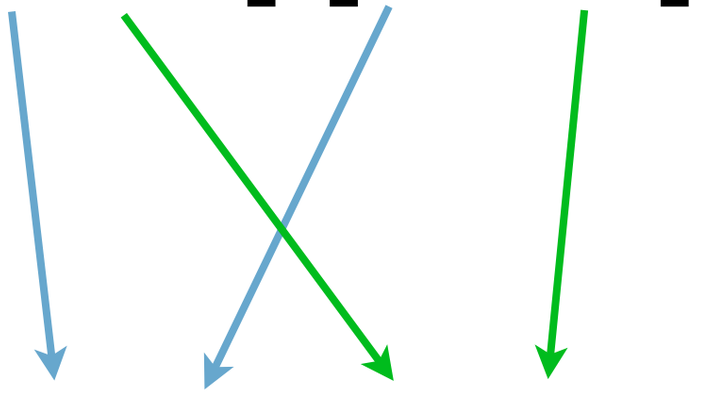
Sample Code

Zip

Take two lists and zip them together such that, corresponding elements of each list are paired together. Pair only until the end of the shorter list.

zip [1, 2, 3] [‘a’ , ’b’]



[(1, ’a’), (2, ’b’)]

```haskell
zip' :: [a] -> [b] -> [(a,b)]
zip' _ [] = []
zip' [] _ = []
zip' (x:xs) (y:ys) = (x,y) : zip' xs ys
```

[ ] = Empty List

_ = Ignore

[x] = List of type x

(a,b) = Pair of type a and b

(head:tail)

Short Lines Only

Make a program that takes some input and prints out only those lines that are shorter than 10 characters.

```haskell
main = interact shortLinesOnly

shortLinesOnly :: String -> String
shortLinesOnly input =
    let allLines = lines input
        shortLines = filter (\line -> length line < 10) allLines
        result = unlines shortLines
    in  result
```

1

```haskell
main = interact (unlines . filter ((<10) . length) . lines)
```

2

```haskell
interact :: (String -> String) -> IO ()
```

```haskell
lines :: String -> [String]
```

```haskell
unlines :: [String] -> String
```

```haskell
(.) :: (b -> c) -> (a -> b) -> a -> c
```

Win with Haskell