$ vs .

```
ghci> :i ($)
($) :: (a -> b) -> a -> b
infixr 0 $
```

```
ghci> :i (.)
(.) :: (b -> c) -> (a -> b) -> a -> c
infixr 9 .
```

$$f \ \$ \ x \equiv f \ x$$

```
f $ x ≡ f x

f $ g $ x ≡ f (g x)
```

```
f $ x ≡ f x

f $ g $ x ≡ f (g x)

f $ g y $ h $ x ≡ f (g y (h x))
```

```
ghci> :i ($)
($) :: (a -> b) -> a -> b
infixr 0 $


ghci> :i (.)
(.) :: (b -> c) -> (a -> b) -> a -> c
infixr 9 .
```

```
f . g ≡ \x -> f(g(x))
```

```
f . g ≡ \x -> f(g(x))

f . g . h ≡ f . (g . h)
```

```
f . g ≡ \x -> f(g(x))

f . g . h ≡ f . (g . h)

f . g y . h ≡ f . ((g y) . h)
```

```
f . g $ x ≡ (f . g) x
```

```
f . g $ x ≡ (f . g) x

f $ g . h ≡ f (g . h)
```

```
f . g $ x ≡ (f . g) x

f $ g . h ≡ f (g . h)

f . g $ h . l $ y ≡ (f . g) ((h . l) y)
```

```
f $ x
```

```
f :: Int -> b
x :: ?
f $ x :: ?
```

```haskell
f $ x
```

```haskell
f :: Int -> b
x :: Int
f $ x :: b
```

f . g

```
f :: String -> b
g :: ?
f . g :: ?
```

f . g

```
f :: String -> b
g :: a -> String
f . g :: a -> b
```

```
f $ g
```

```haskell
f :: ?
g :: a -> String
f $ g :: Int
```

```
f $ g
```

```
f :: (a -> String) -> Int
g :: a -> String
f $ g :: Int
```

```
f $ g
```

```
f :: ?
g :: Int
f $ g :: ([a] -> a)
```

```
f $ g
```

```
f :: Int -> [a] -> a
g :: Int
f $ g :: ([a] -> a)
```

```
f 5 . g
```

```
f :: Int -> String -> Bool
g :: ?
f 5 . g :: ?
```

```
f 5 . g
```

```
f :: Int -> String -> Bool
g :: a -> String
f 5 . g :: a -> Bool
```

```
f 5 $ x
```

```
f :: Int -> String -> Bool
x :: ?
f 5 $ x :: ?
```

```
f 5 $ x
```

```
f :: Int -> String -> Bool
x :: String
f 5 $ x :: Bool
```

```
f 5 . g $ h “foo”
```

```
f :: ?
g :: ?
h :: ?
f 5 . g $ h “foo” :: ?
```

```
f 5 . g $ h “foo”
```

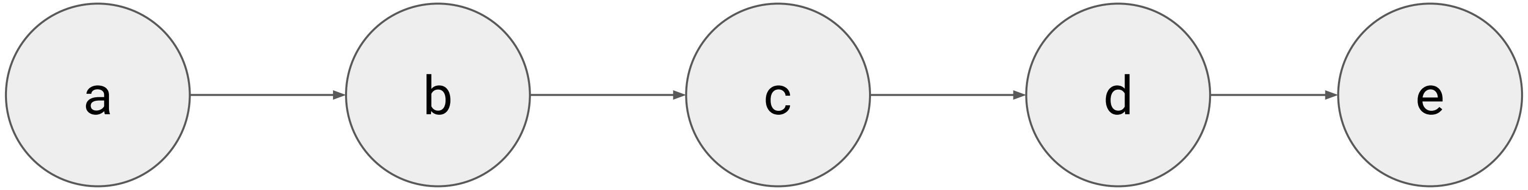
```
f :: Num d => d -> b -> c
g :: a -> b
h :: String -> a
f 5 . g $ h “foo” :: c
```

map

map f list

```haskell
map :: (a -> b) -> [a] -> [b]
map _ [] = []
map f (x:xs) = f x : map f xs
```

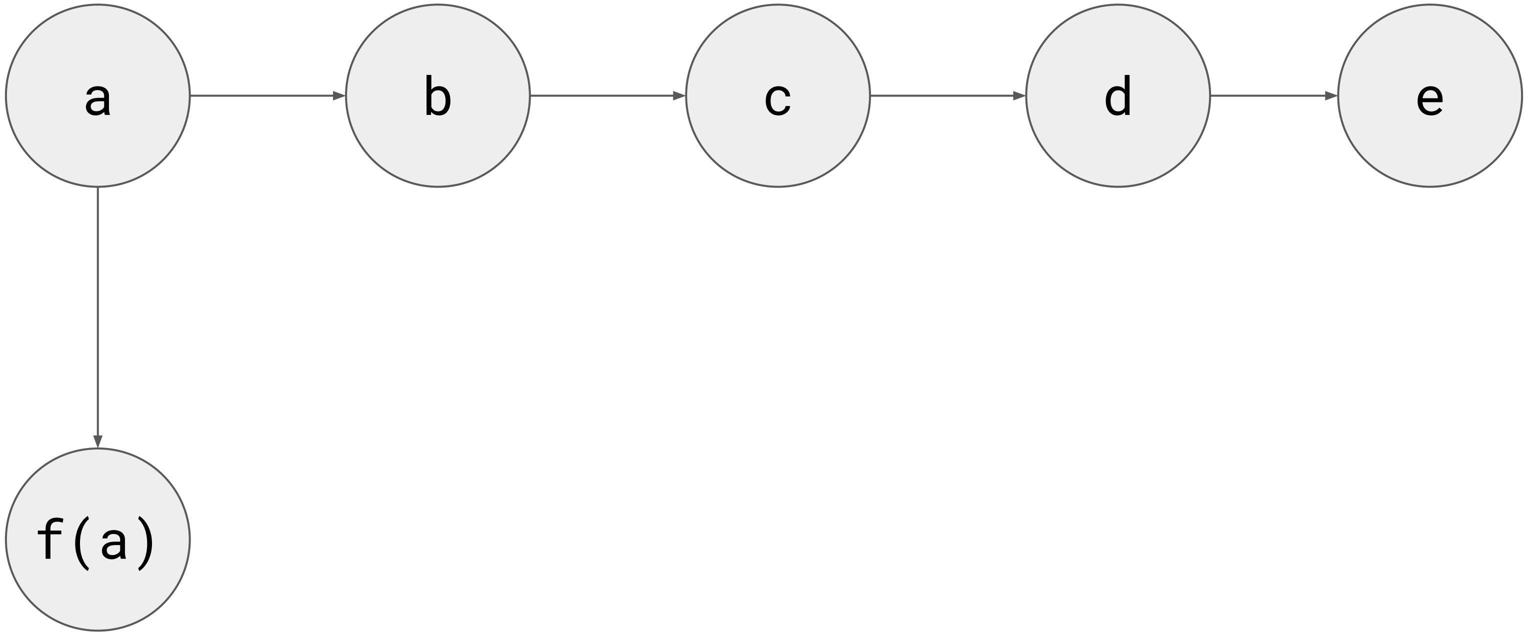
[a, b, c, d, e]


$$\begin{tikzcd} a \arrow[r] & b \arrow[r] & c \arrow[r] & d \arrow[r] & e \end{tikzcd}$$

`map f [a, b, c, d, e]`

$$\begin{tikzcd}
a \arrow[r] & b \arrow[r] & c \arrow[r] & d \arrow[r] & e
\end{tikzcd}$$

```
map f [a, b, c, d, e]
```



map f [a, b, c, d, e]

$$\begin{tikzcd}
a \arrow[r] \arrow[d] & b \arrow[r] \arrow[d] & c \arrow[r] & d \arrow[r] & e \\
f(a) \arrow[r] & f(b) & & &
\end{tikzcd}$$

`map f [a, b, c, d, e]`

$$\begin{tikzcd}
a \arrow[r] \arrow[d] & b \arrow[r] \arrow[d] & c \arrow[r] \arrow[d] & d \arrow[r] & e \\
f(a) \arrow[r] & f(b) \arrow[r] & f(c) & &
\end{tikzcd}$$

map f [a, b, c, d, e]

$$\begin{tikzcd}
a \arrow[r] \arrow[d] & b \arrow[r] \arrow[d] & c \arrow[r] \arrow[d] & d \arrow[r] \arrow[d] & e \\
f(a) \arrow[r] & f(b) \arrow[r] & f(c) \arrow[r] & f(d) &
\end{tikzcd}$$

`map f [a, b, c, d, e]`

$$\begin{tikzcd}
a \arrow[r] \arrow[d] & b \arrow[r] \arrow[d] & c \arrow[r] \arrow[d] & d \arrow[r] \arrow[d] & e \arrow[d] \\
f(a) \arrow[r] & f(b) \arrow[r] & f(c) \arrow[r] & f(d) \arrow[r] & f(e)
\end{tikzcd}$$

```
map (*2) [1, 2, 3, 4, 5]
```

$$\begin{tikzcd}
1 \arrow[r] \arrow[d] & 2 \arrow[r] \arrow[d] & 3 \arrow[r] \arrow[d] & 4 \arrow[r] \arrow[d] & 5 \arrow[d] \\
2 \arrow[r] & 4 \arrow[r] & 6 \arrow[r] & 8 \arrow[r] & 10
\end{tikzcd}$$

`map even [1, 2, 3, 4, 5]`

$$\begin{tikzcd}
1 \arrow[r] \arrow[d] & 2 \arrow[r] \arrow[d] & 3 \arrow[r] \arrow[d] & 4 \arrow[r] \arrow[d] & 5 \arrow[d] \\
\text{False} \arrow[r] & \text{True} \arrow[r] & \text{False} \arrow[r] & \text{True} \arrow[r] & \text{False}
\end{tikzcd}$$

`filter`

filter f list

```haskell
filter :: (a -> Bool) -> [a] -> [a]
filter _ [] = []
filter p (x:xs)
  | p x = x : filter p xs
  | otherwise = filter p xs
```

[a, b, c, d, e]

$$\begin{tikzcd}
a \arrow[r] & b \arrow[r] & c \arrow[r] & d \arrow[r] & e
\end{tikzcd}$$

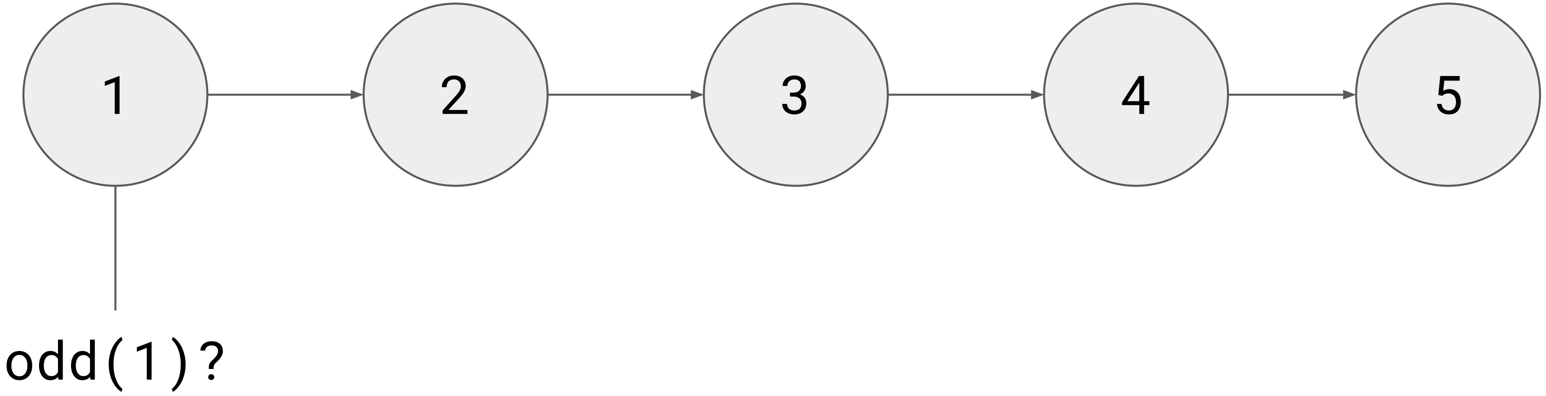
filter f [a, b, c, d, e]

$$\begin{tikzcd}
a \arrow[r] \arrow[d, "f(a)?"] & b \arrow[r] \arrow[d, "f(b)?"] & c \arrow[r] \arrow[d, "f(c)?"] & d \arrow[r] \arrow[d, "f(d)?"] & e \arrow[d, "f(e)?"] \\
a \arrow[r] & b \arrow[r] & c \arrow[r] & d \arrow[r] & e
\end{tikzcd}$$

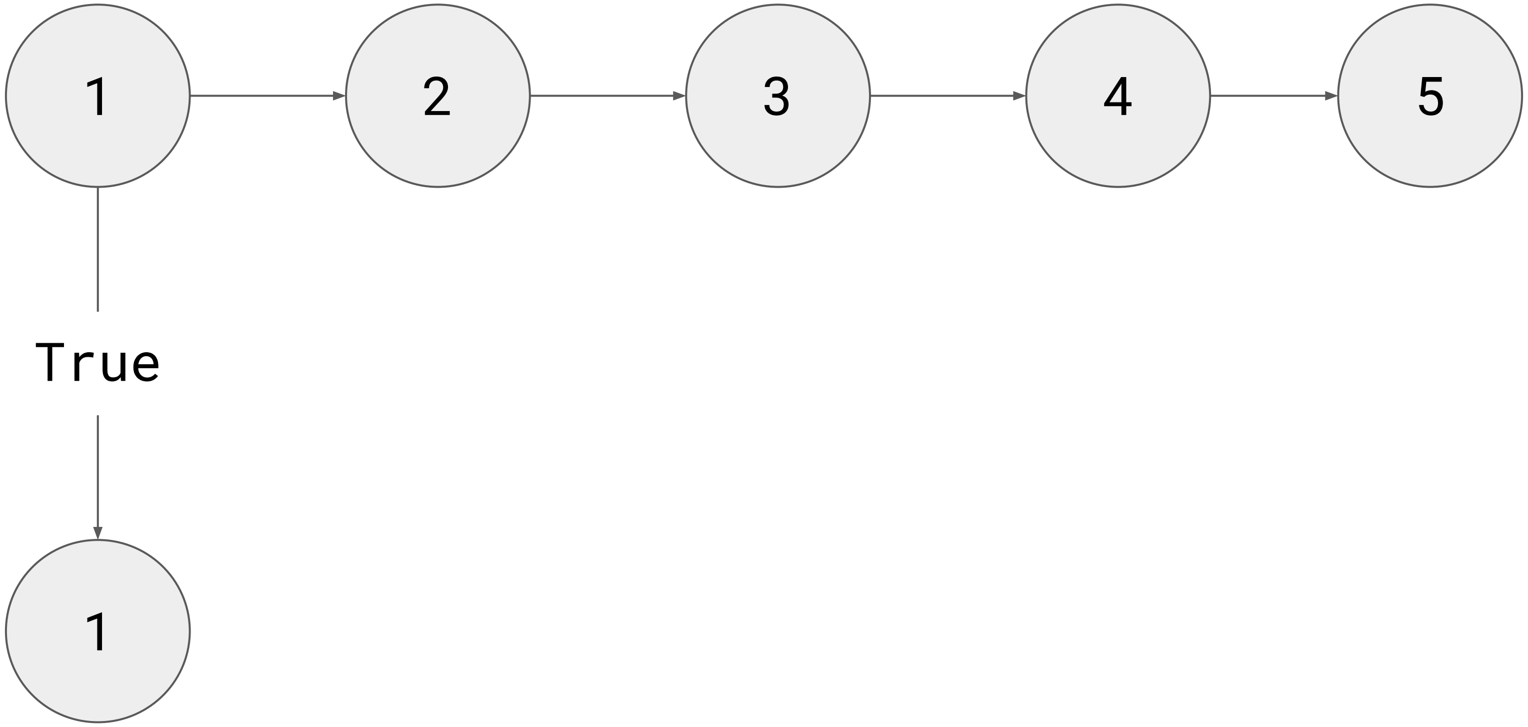
`filter odd [1, 2, 3, 4, 5]`

$$\begin{tikzcd} 1 \arrow[r] & 2 \arrow[r] & 3 \arrow[r] & 4 \arrow[r] & 5 \end{tikzcd}$$

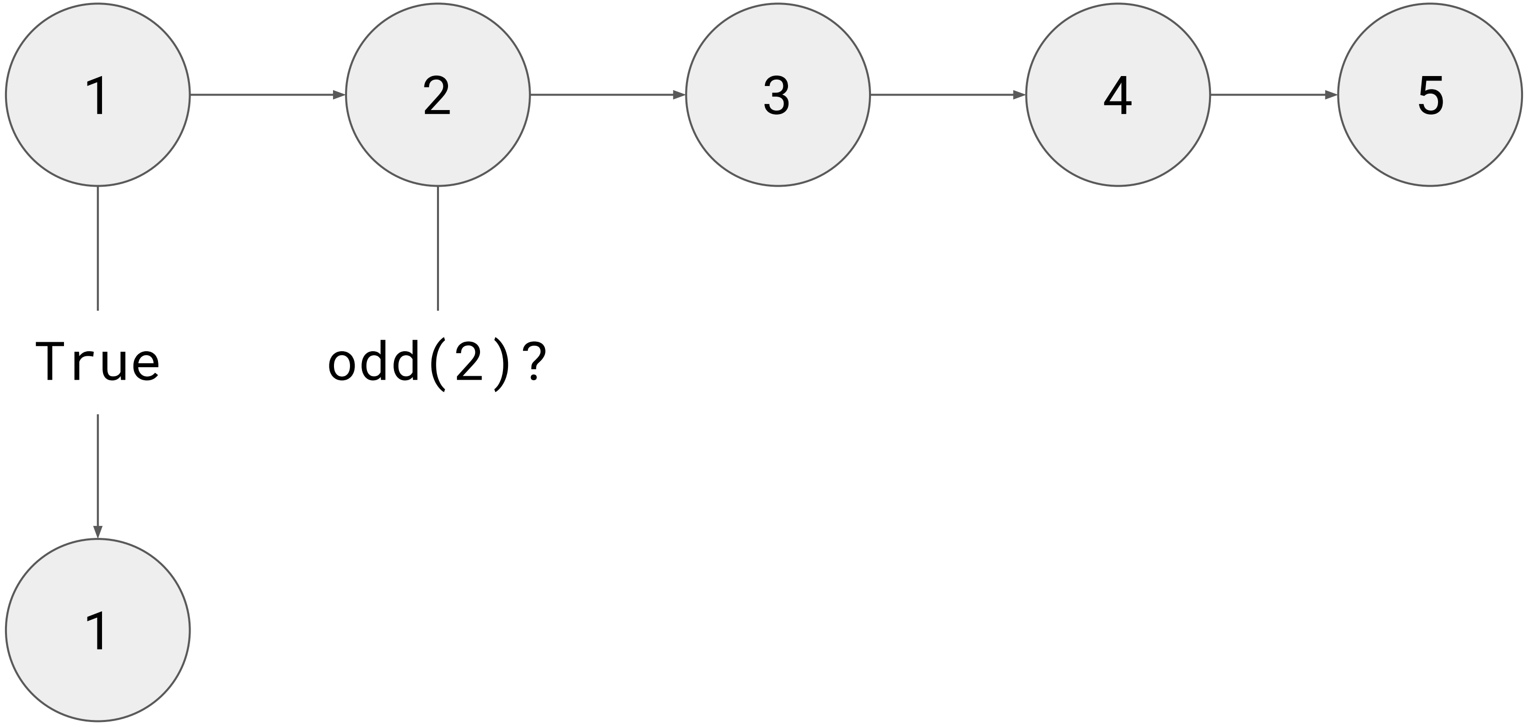
filter odd [1, 2, 3, 4, 5]



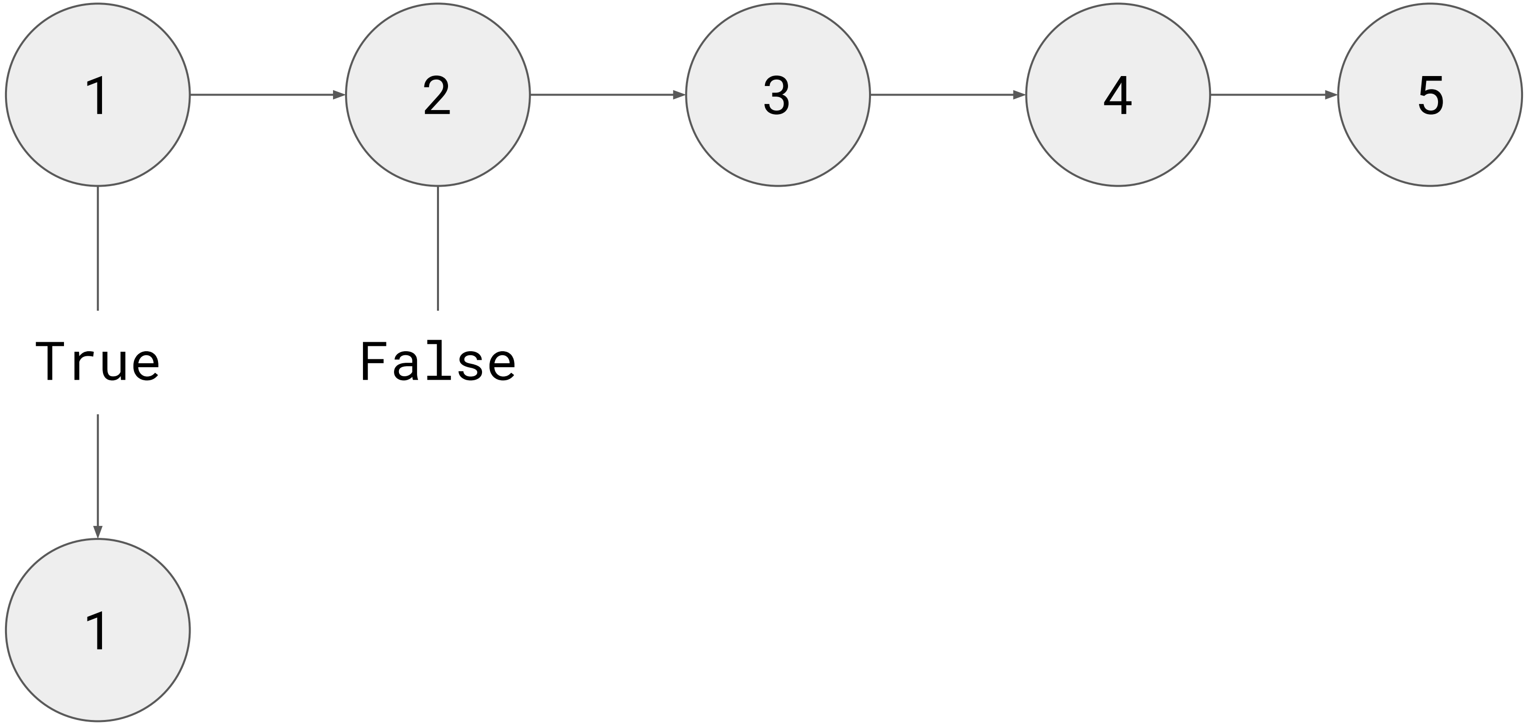
```
filter odd [1, 2, 3, 4, 5]
```



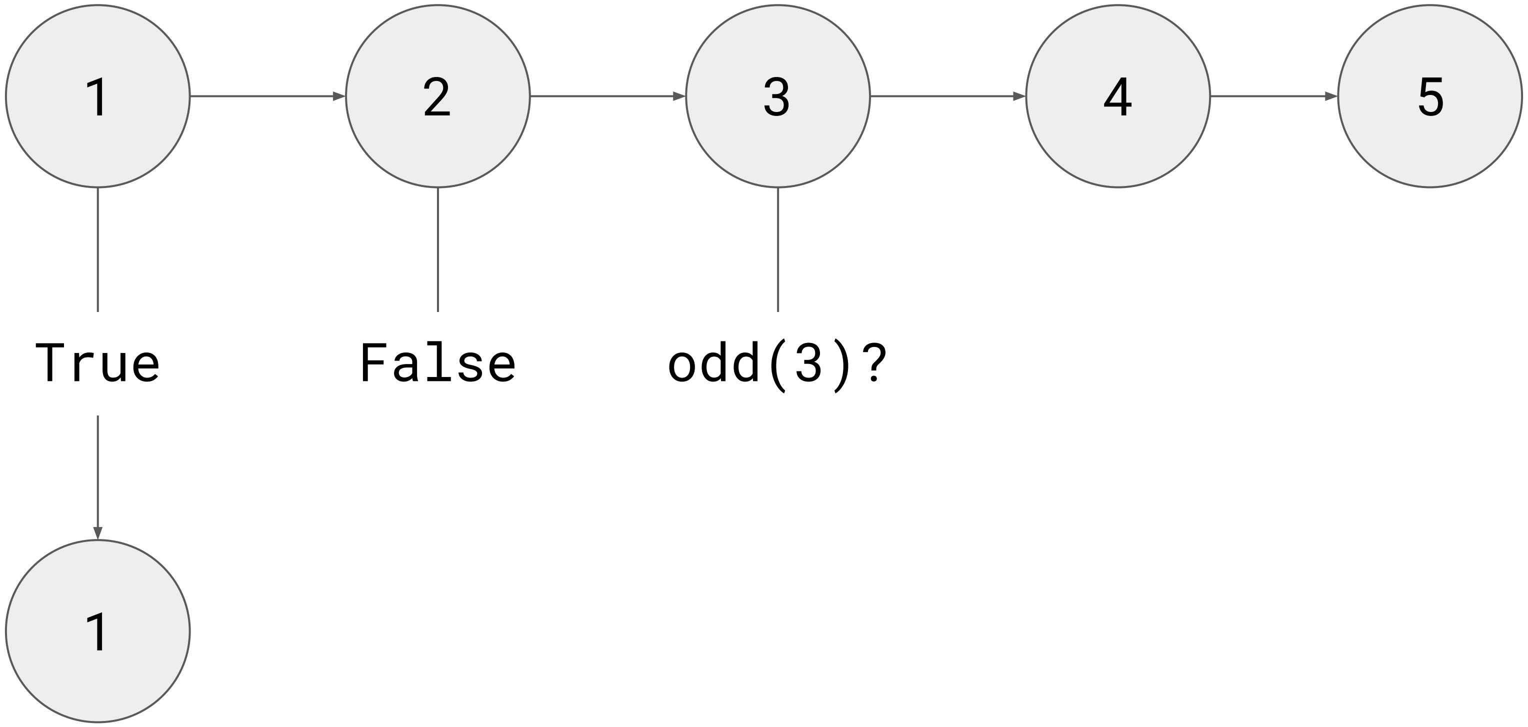
`filter odd [1, 2, 3, 4, 5]`



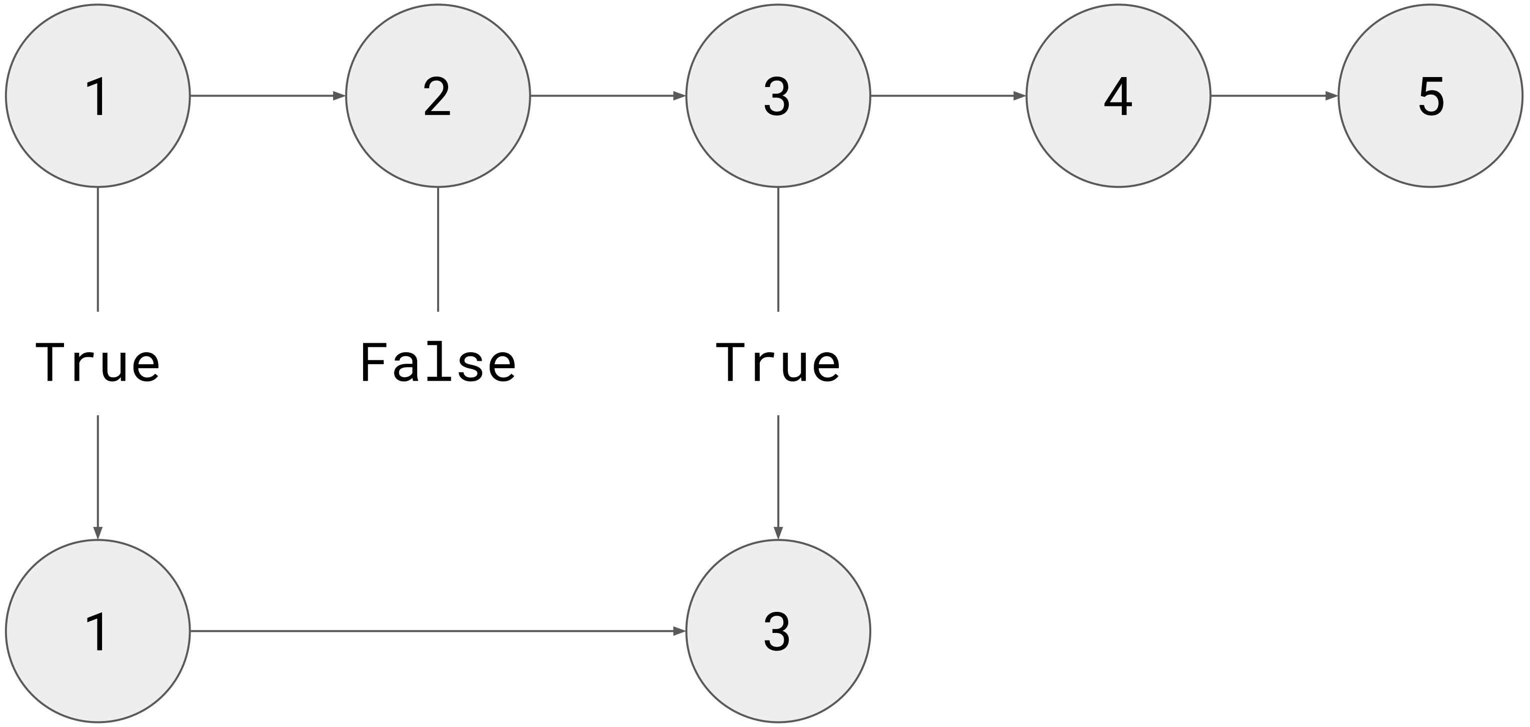
```
filter odd [1, 2, 3, 4, 5]
```



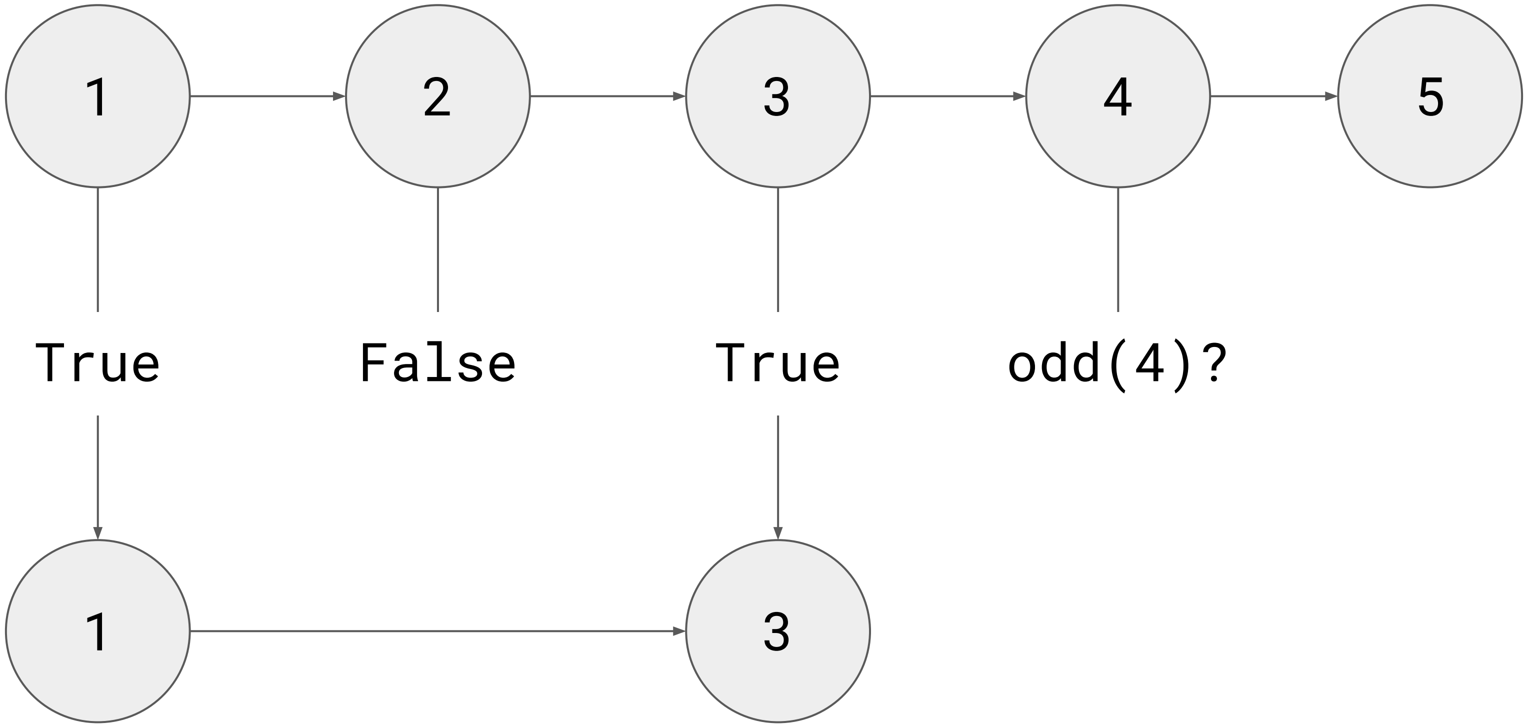
```
filter odd [1, 2, 3, 4, 5]
```



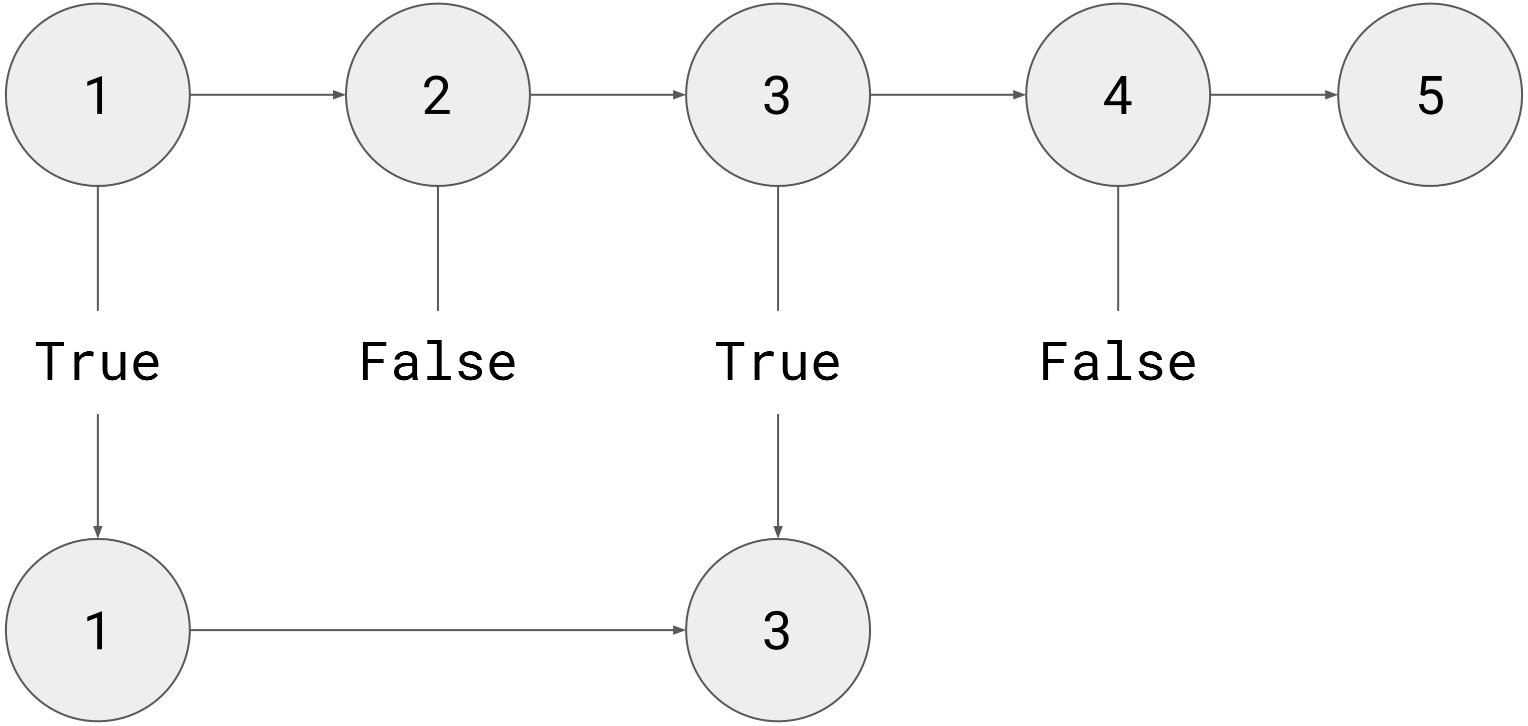
`filter odd [1, 2, 3, 4, 5]`



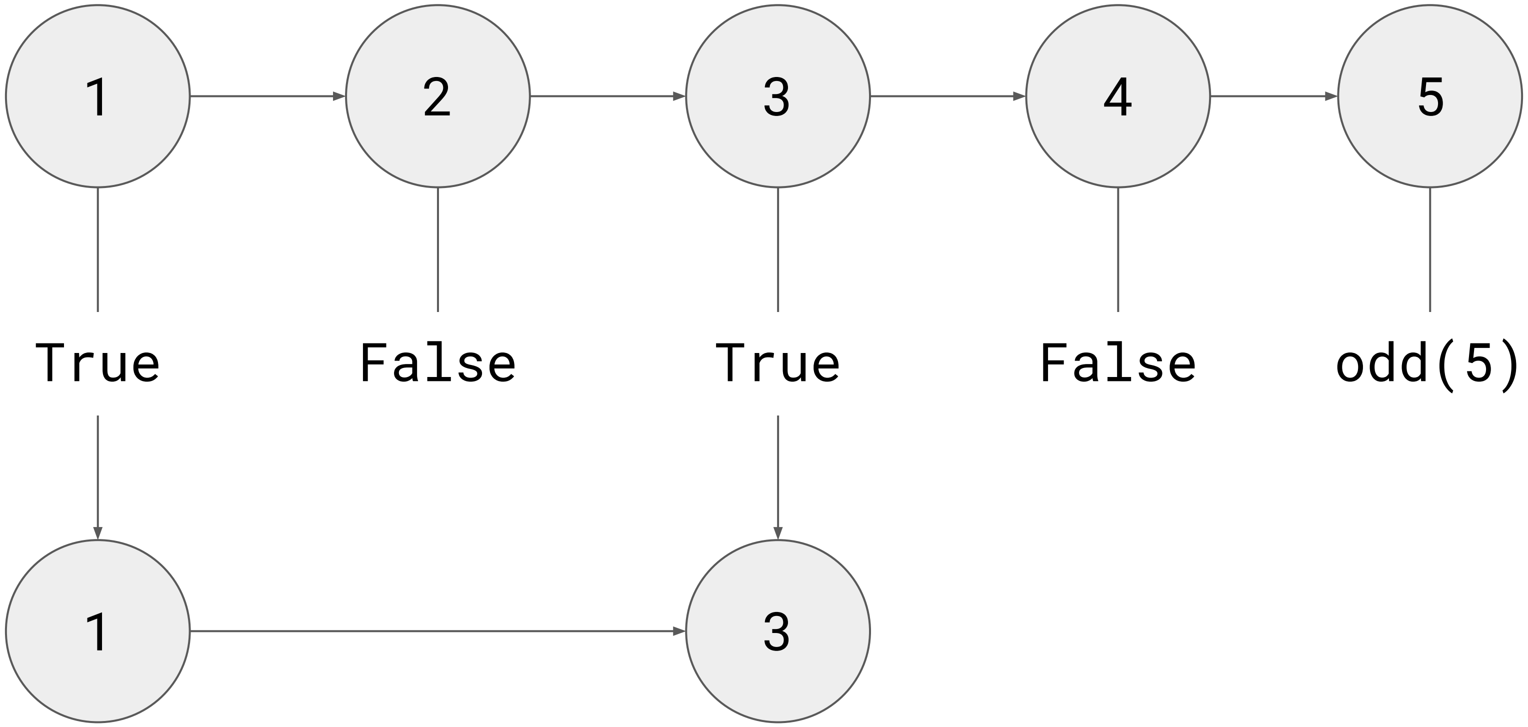
filter odd [1, 2, 3, 4, 5]



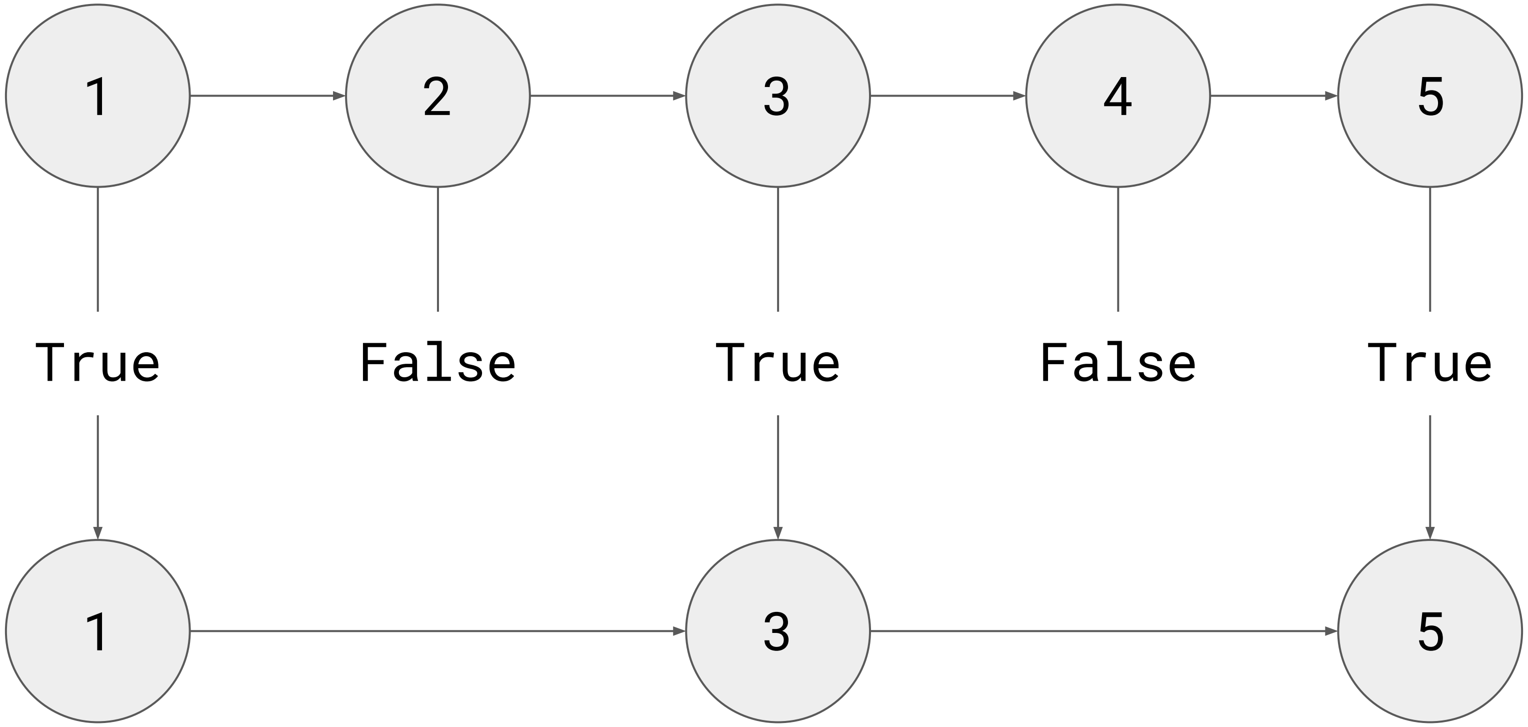
```
filter odd [1, 2, 3, 4, 5]
```



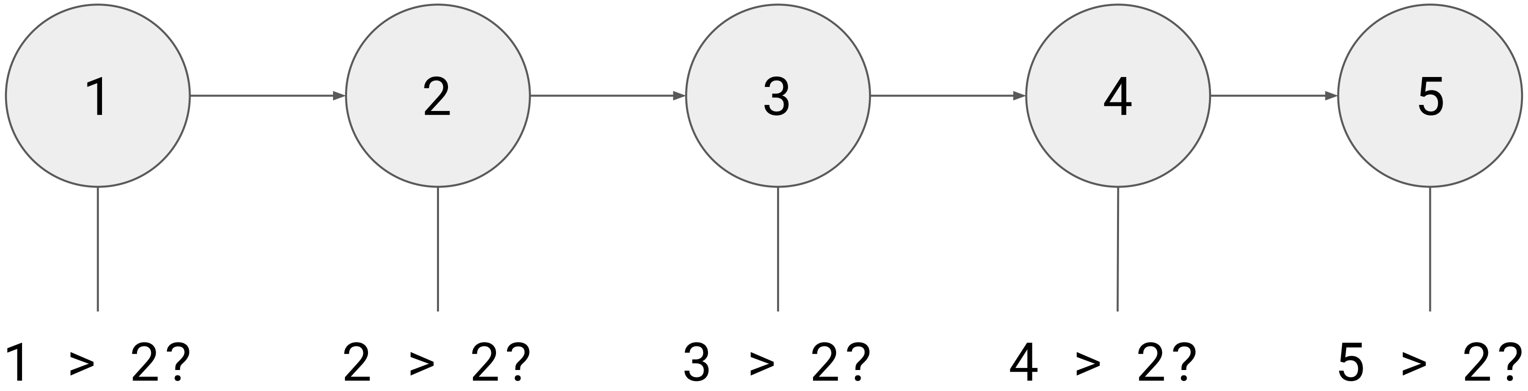
filter odd [1, 2, 3, 4, 5]



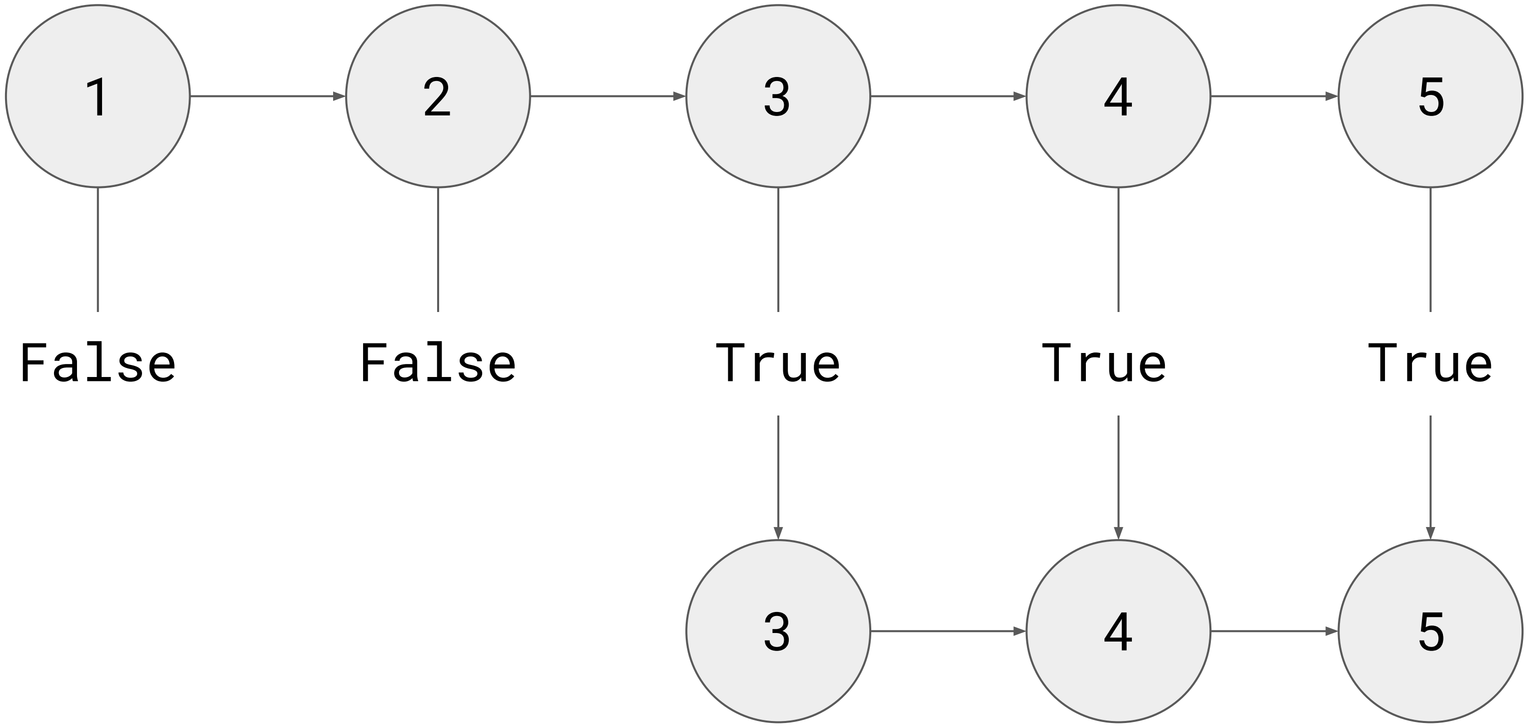
`filter odd [1, 2, 3, 4, 5]`



```
filter (>2) [a, b, c, d, e]
```



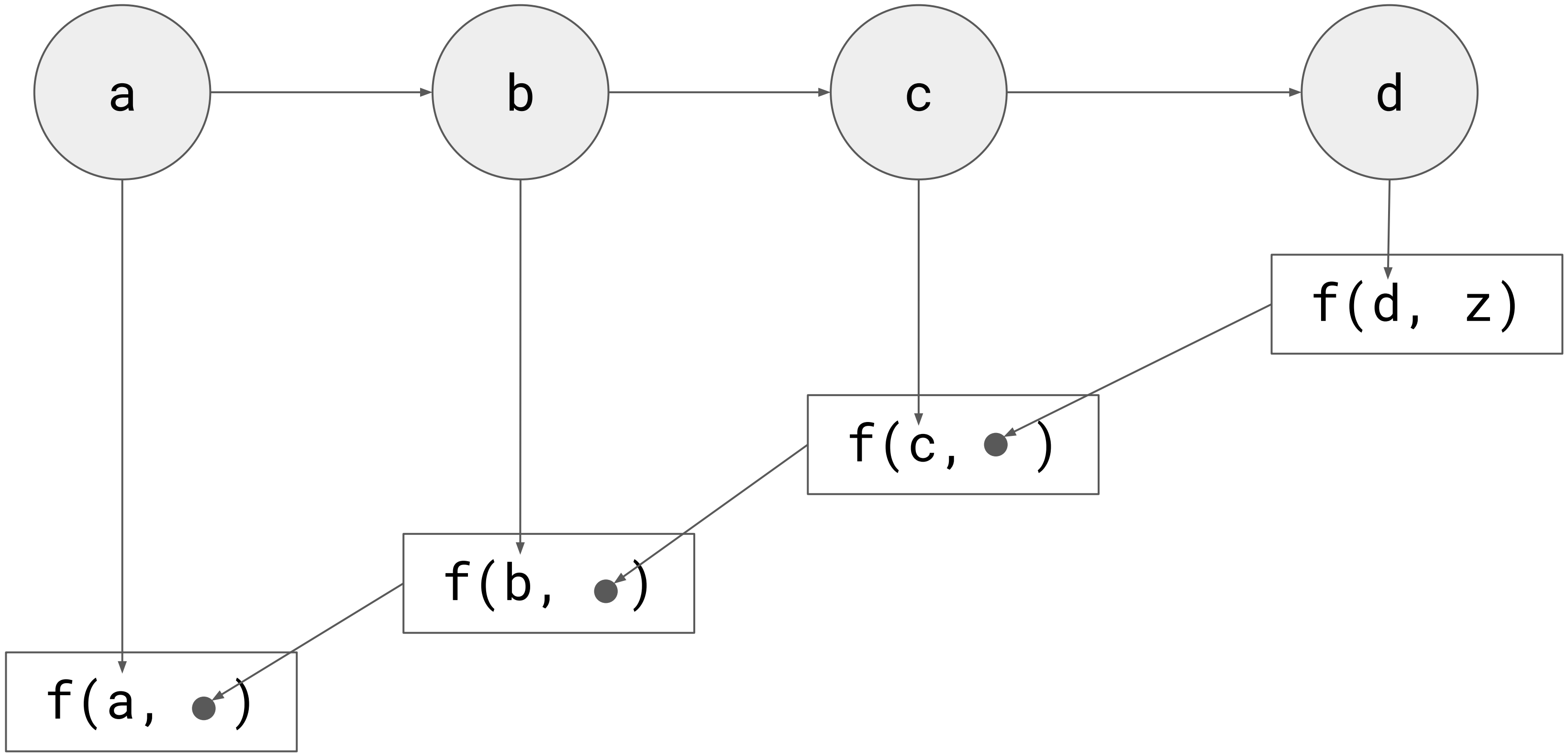
```
filter (>2) [a, b, c, d, e]
```



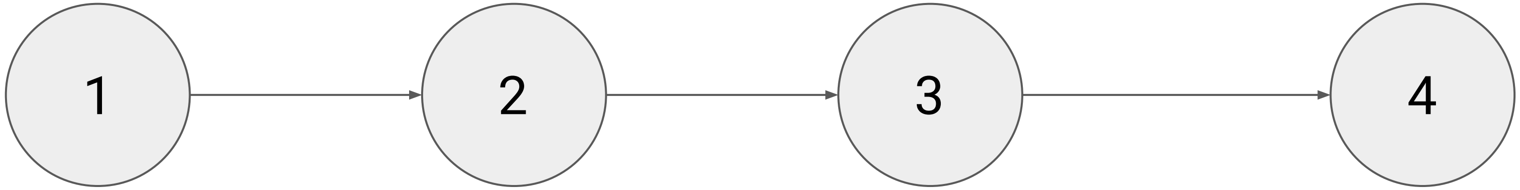
`foldr`

foldr f z list

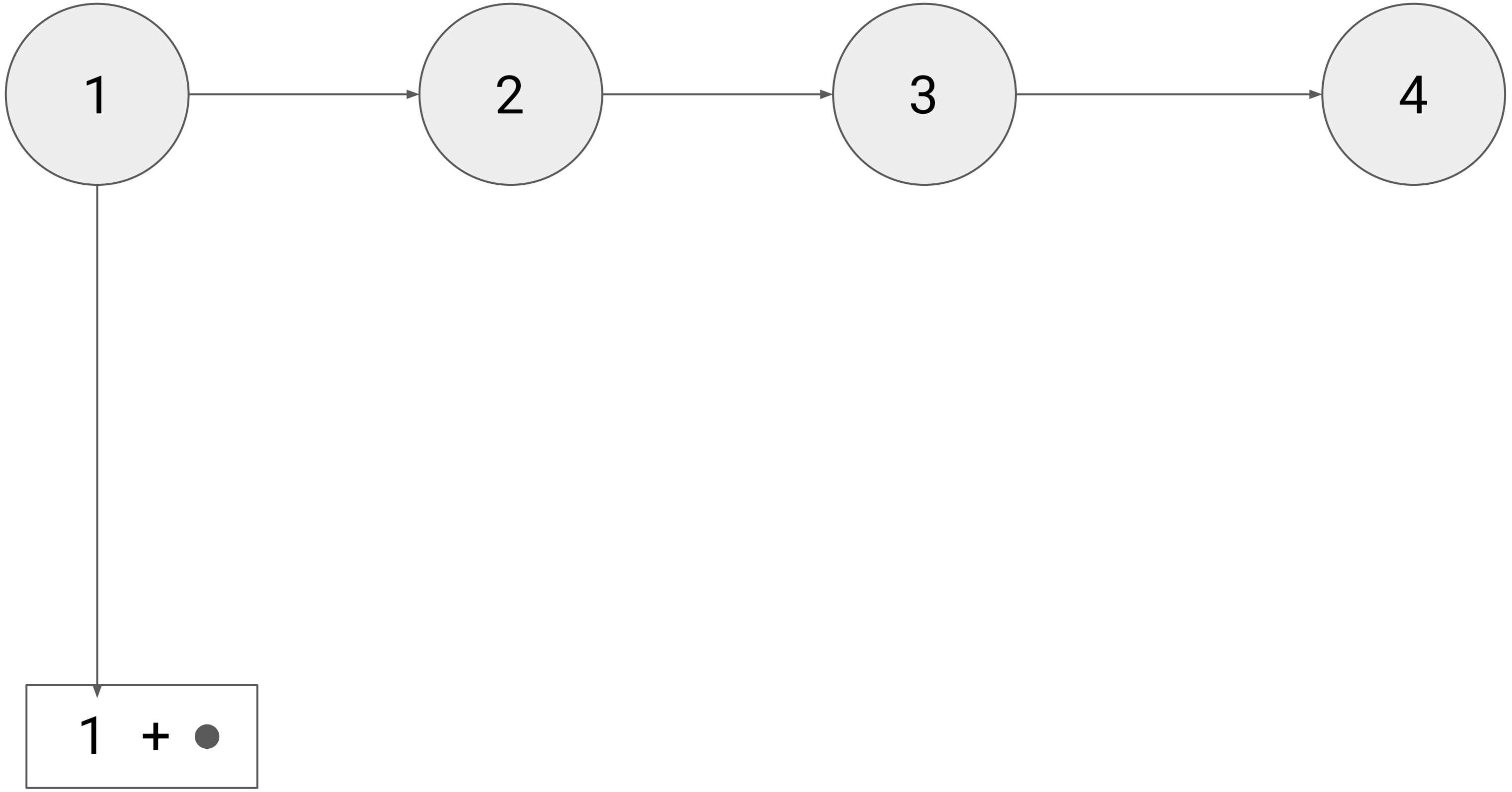
```haskell
foldr :: (a -> b -> b) -> b -> [a] -> b
foldr f z []     = z
foldr f z (x:xs) = x `f` (foldr f z xs)
```



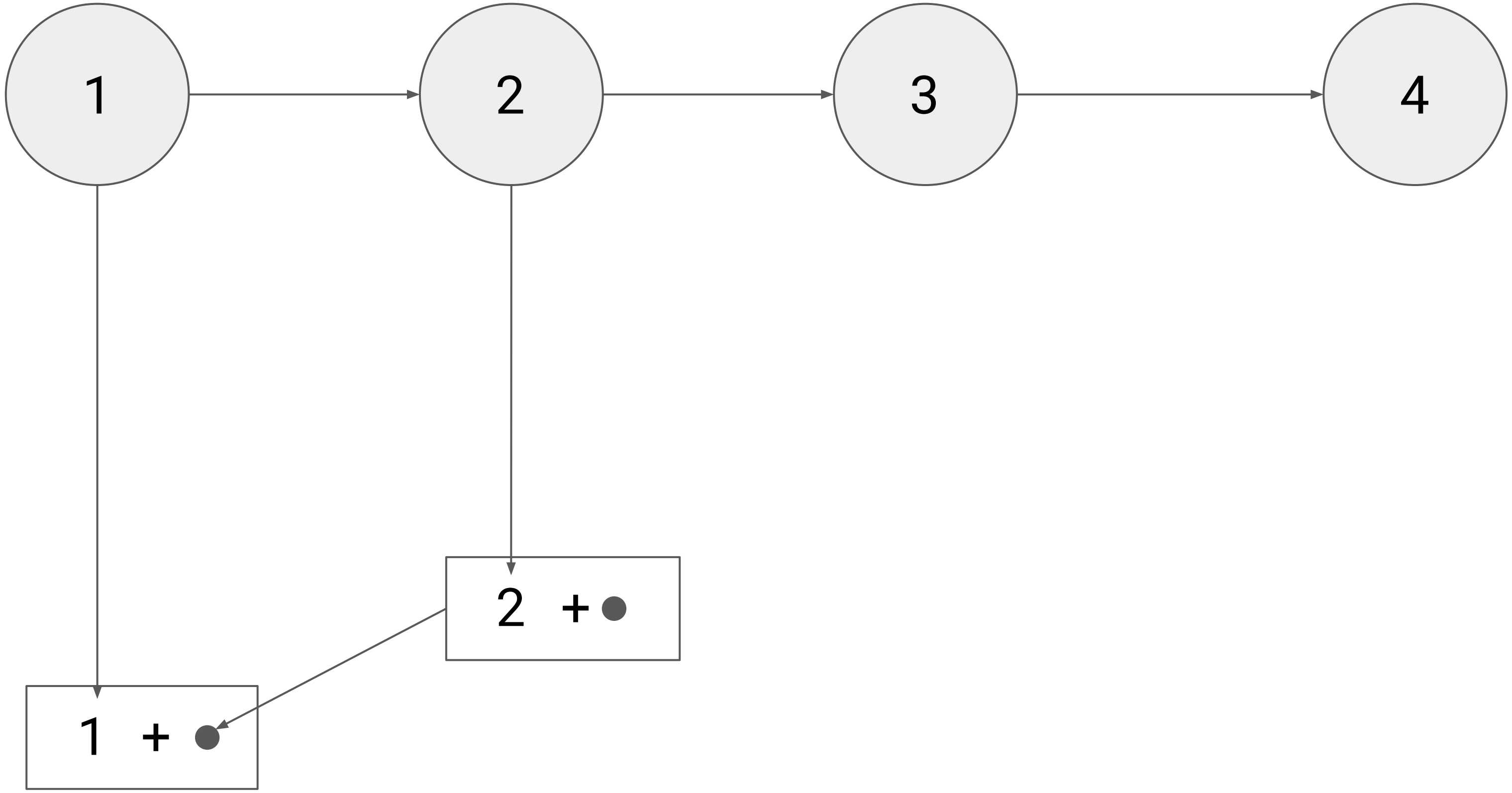
```
foldr f z [a, b, c, d]
```



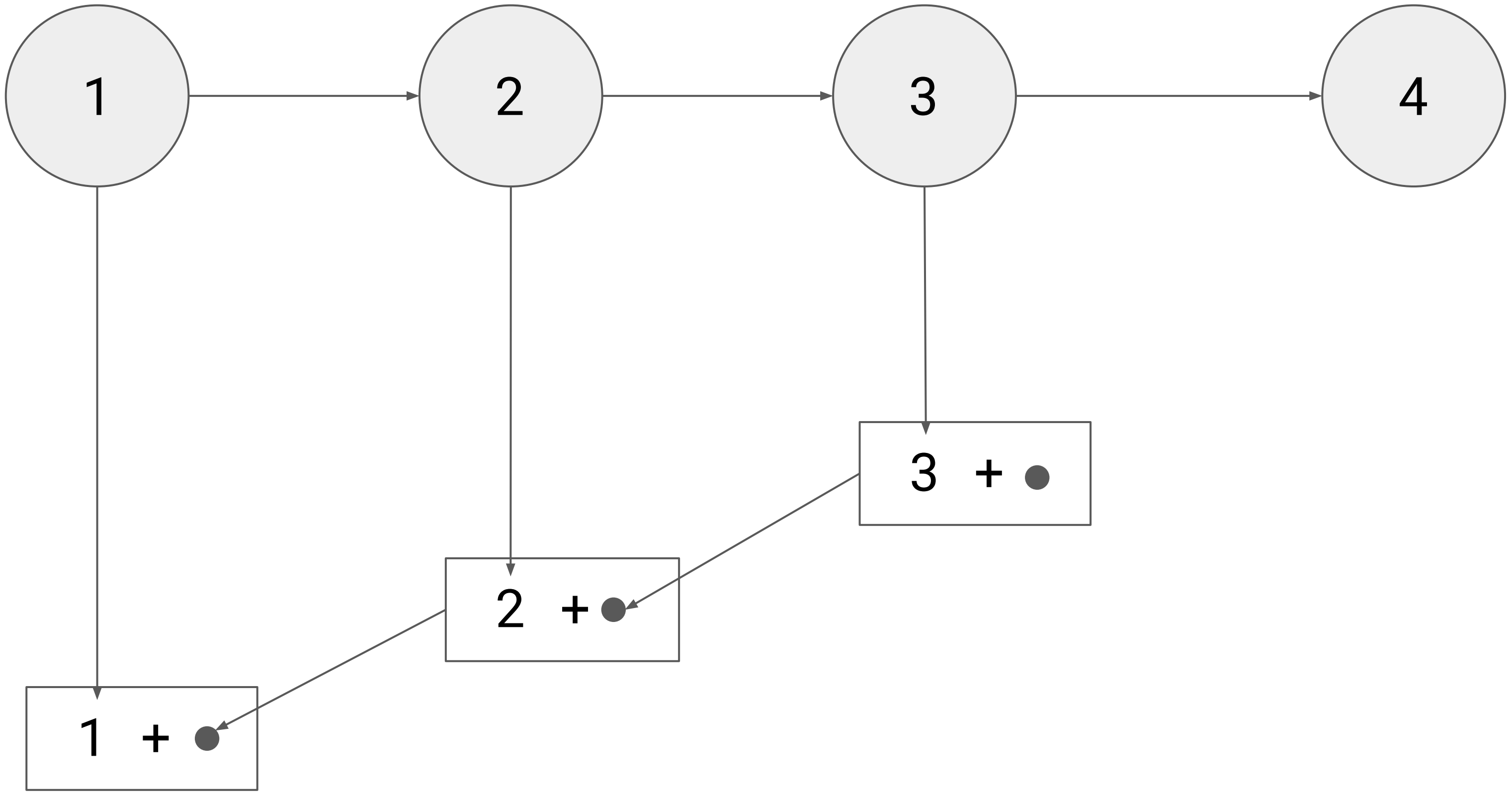
```
foldr (+) 0 [1, 2, 3, 4]
```



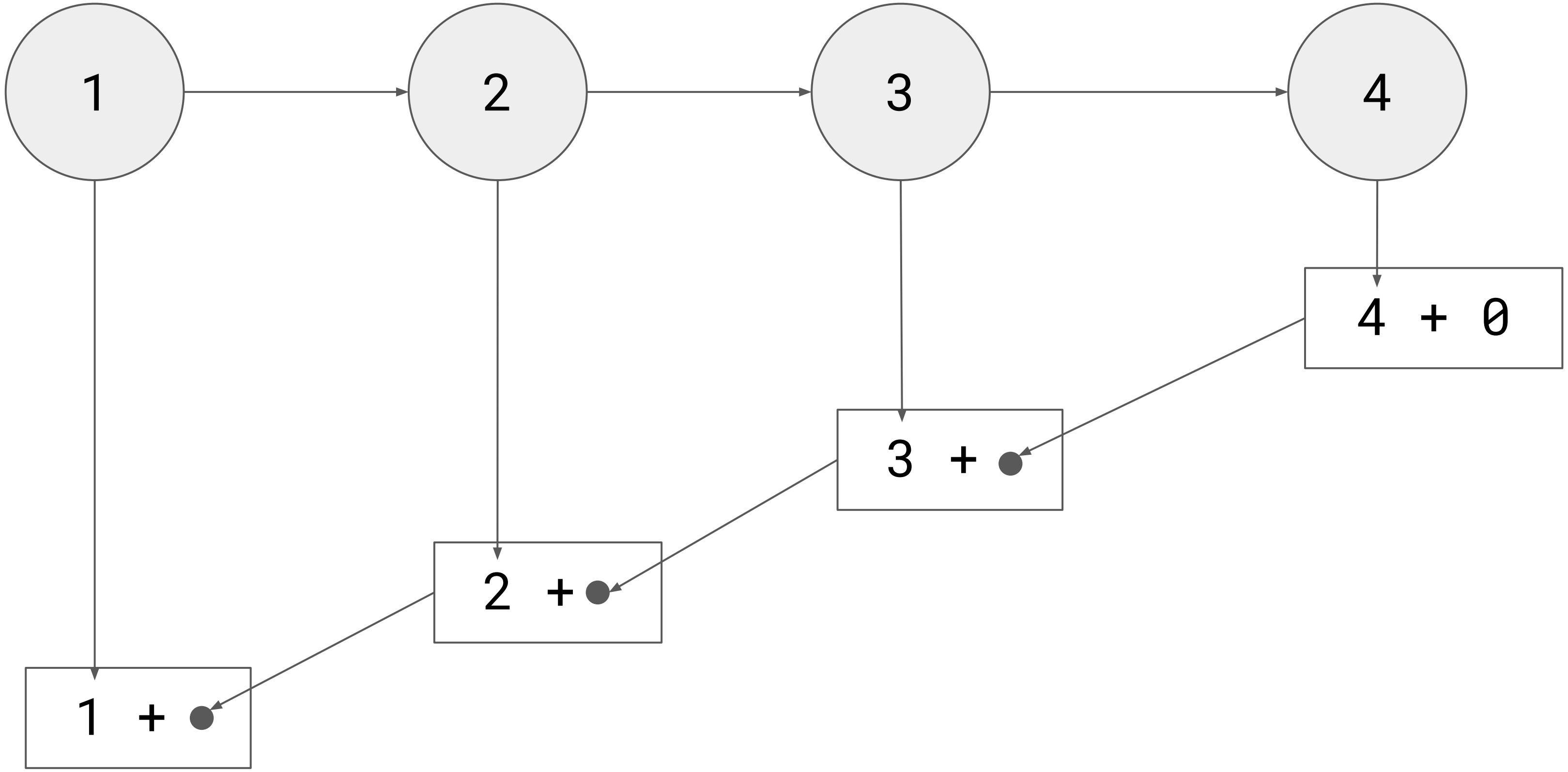
```
foldr (+) 0 [1, 2, 3, 4]
```



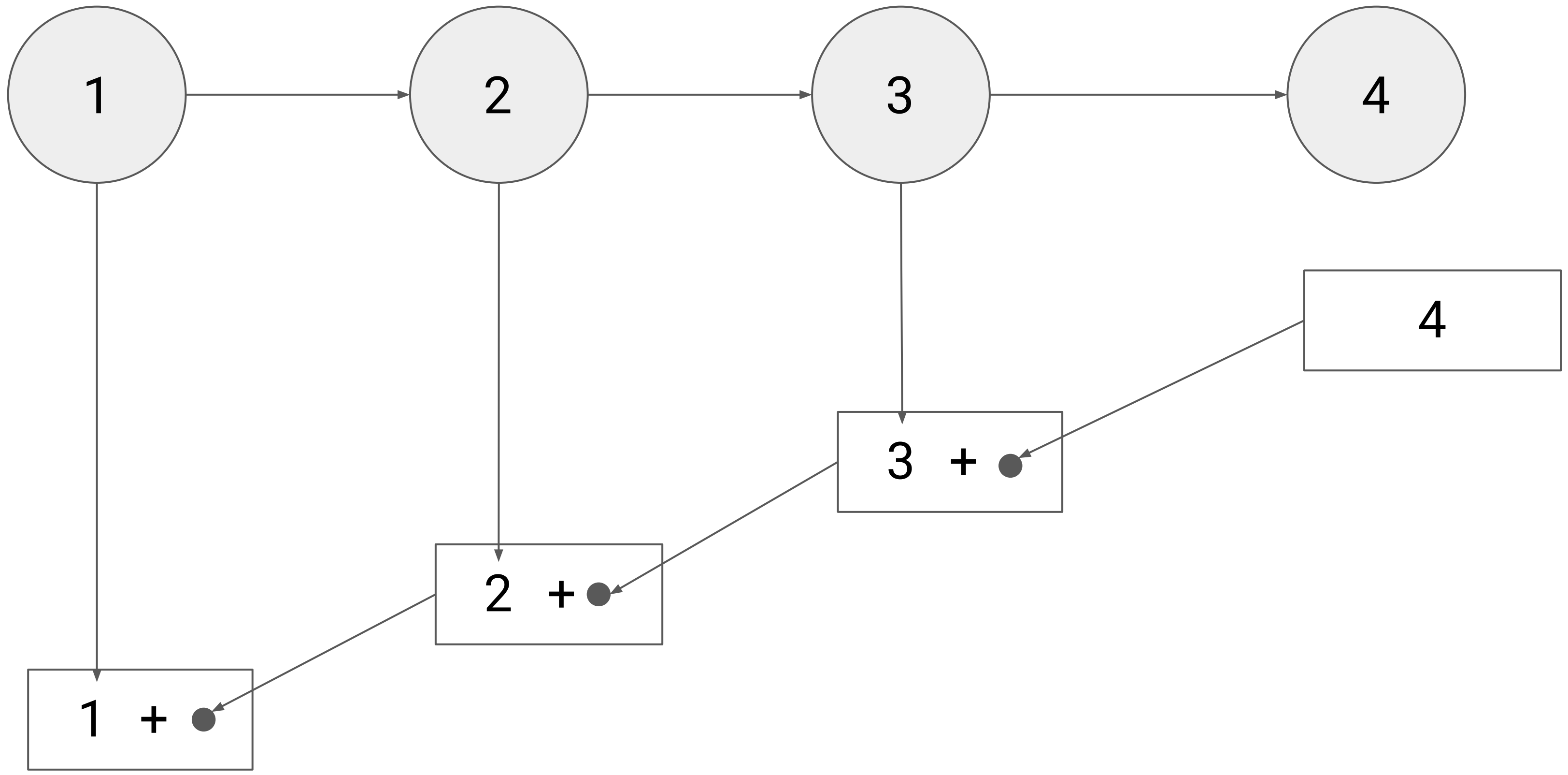
```
foldr (+) 0 [1, 2, 3, 4]
```



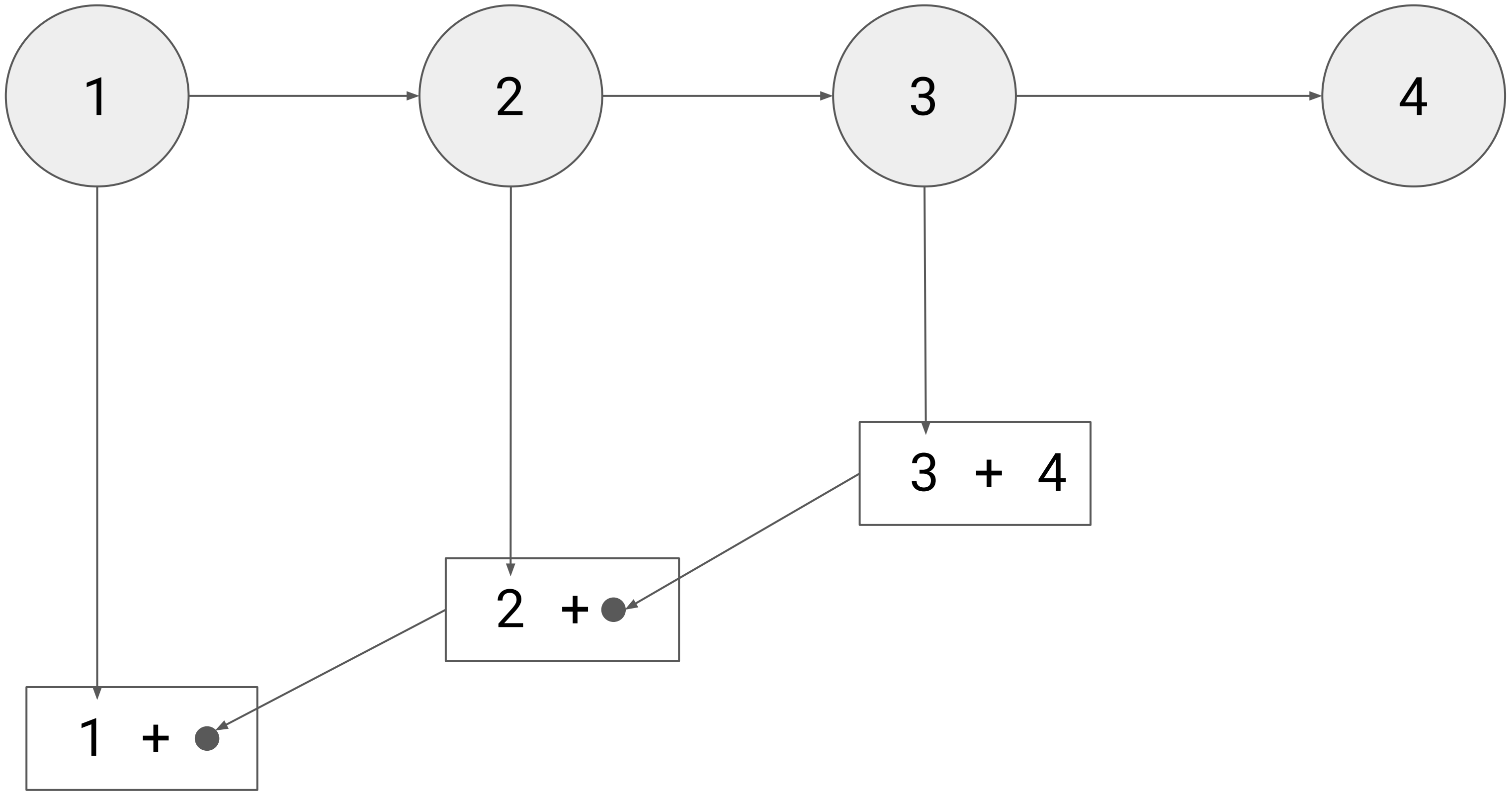
```
foldr (+) 0 [1, 2, 3, 4]
```



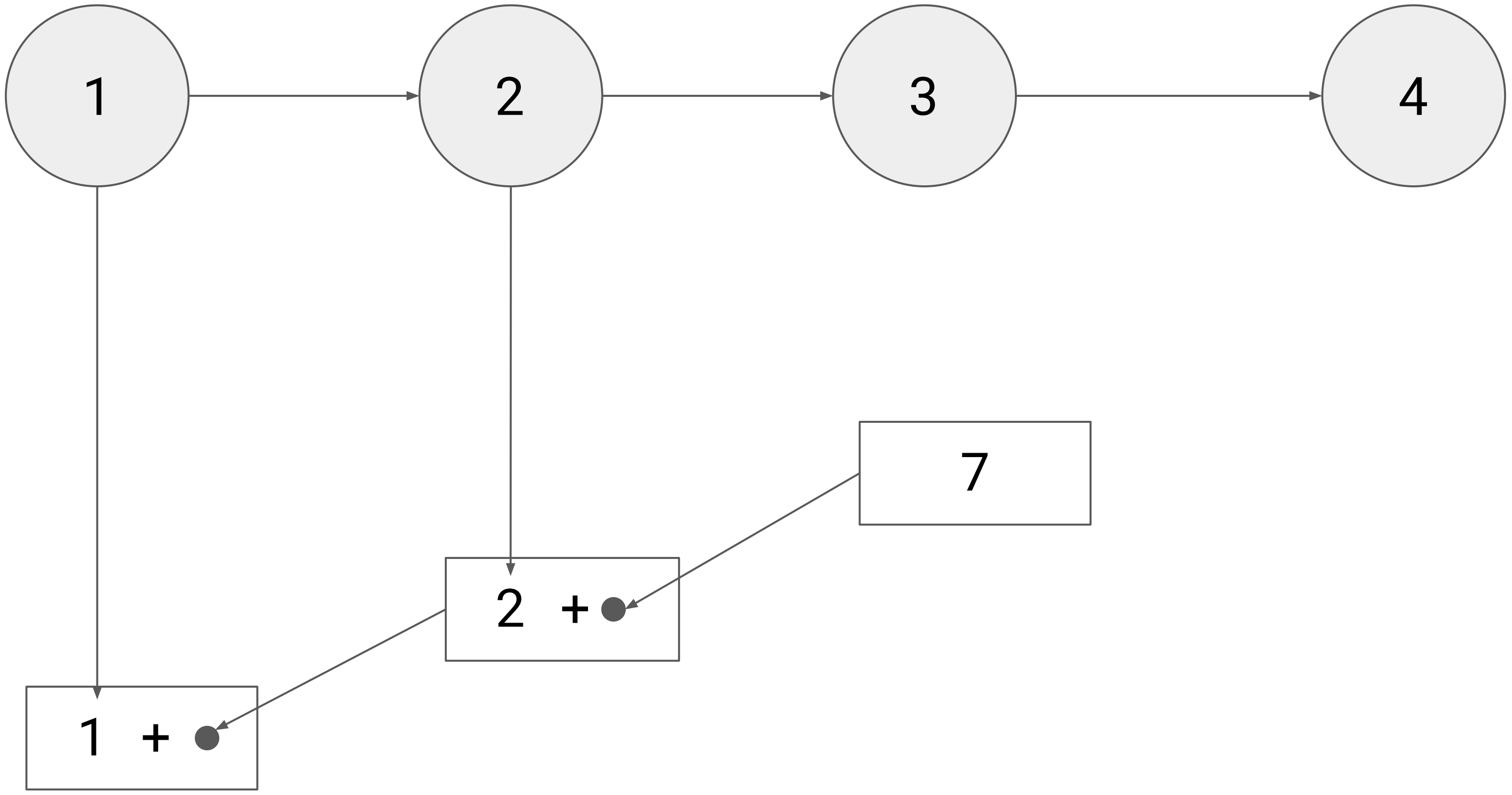
```
foldr (+) 0 [1, 2, 3, 4]
```



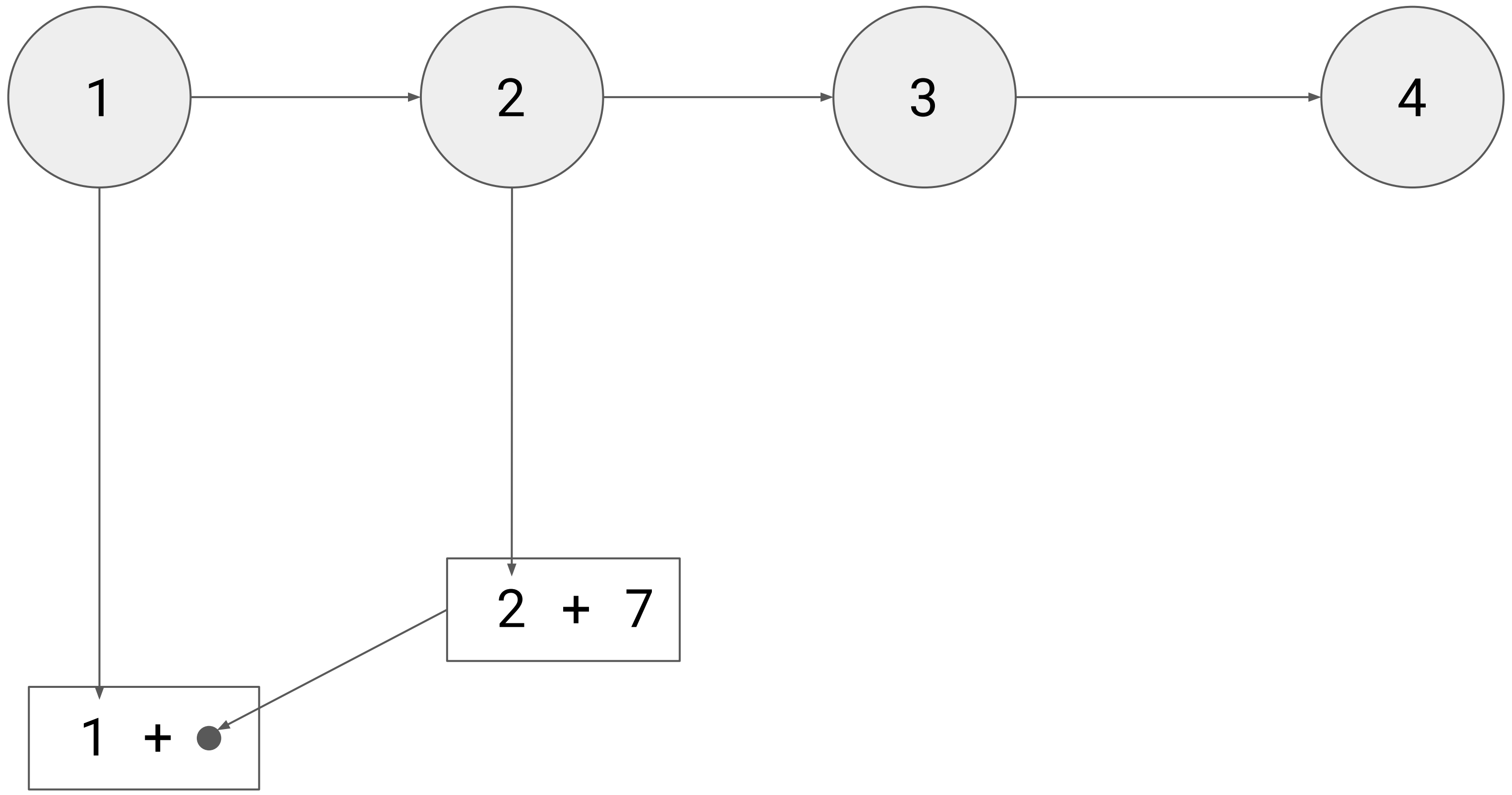
```
foldr (+) 0 [1, 2, 3, 4]
```



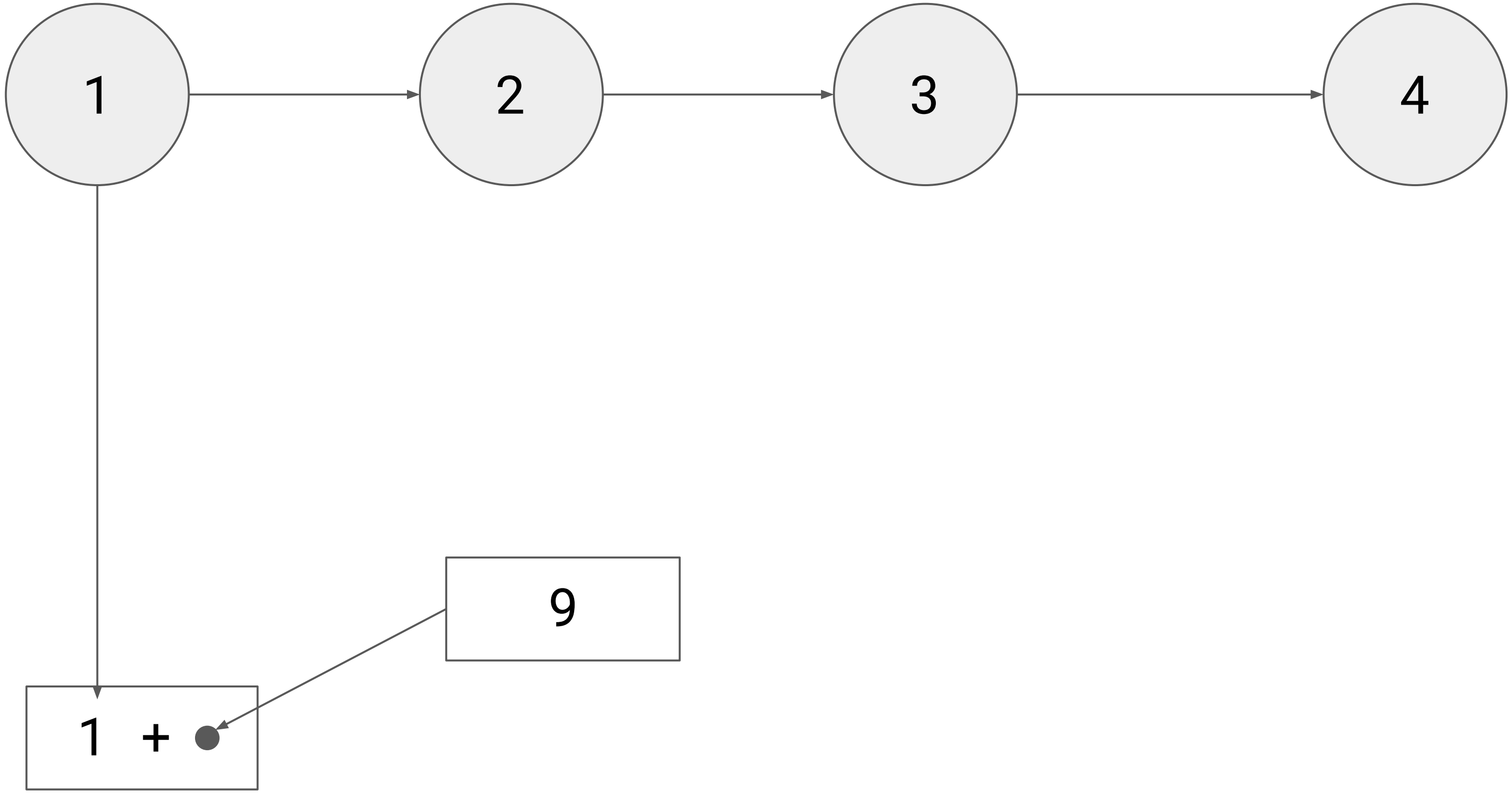
```
foldr (+) 0 [1, 2, 3, 4]
```



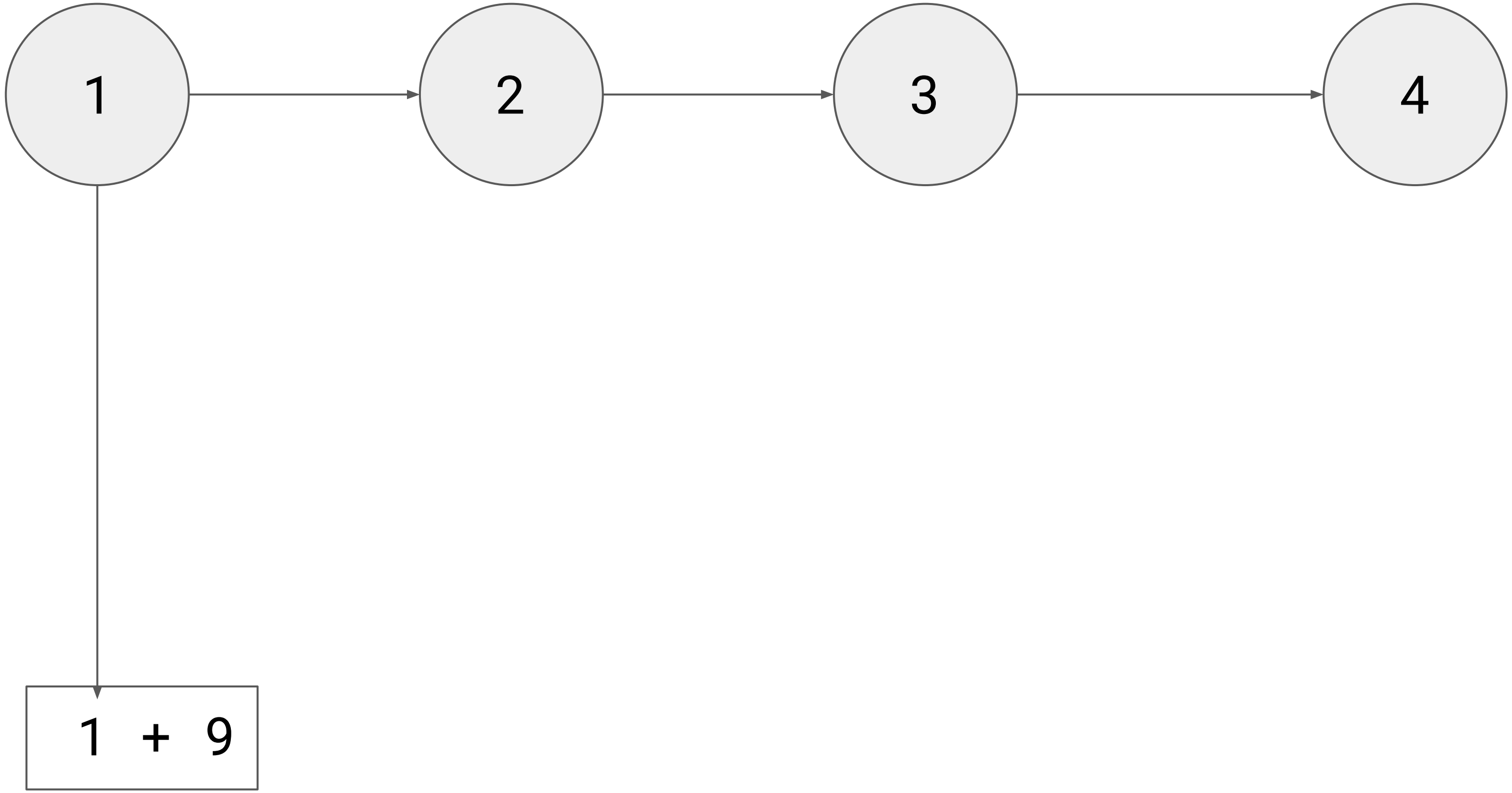
```
foldr (+) 0 [1, 2, 3, 4]
```



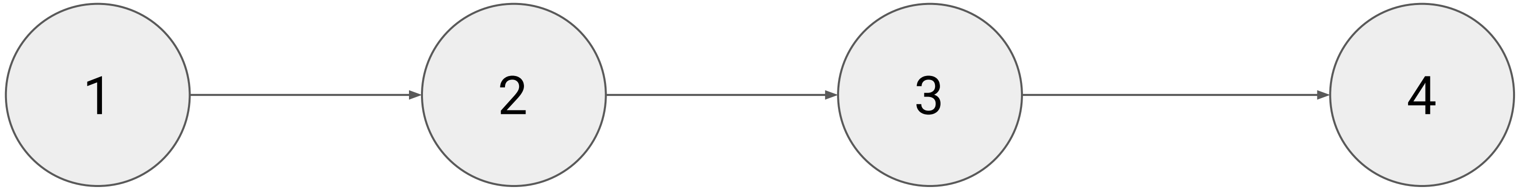
```
foldr (+) 0 [1, 2, 3, 4]
```

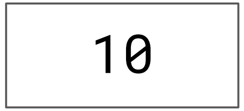


```
foldr (+) 0 [1, 2, 3, 4]
```



```
foldr (+) 0 [1, 2, 3, 4]
```





```
foldr (+) 0 [1, 2, 3, 4]
```

end