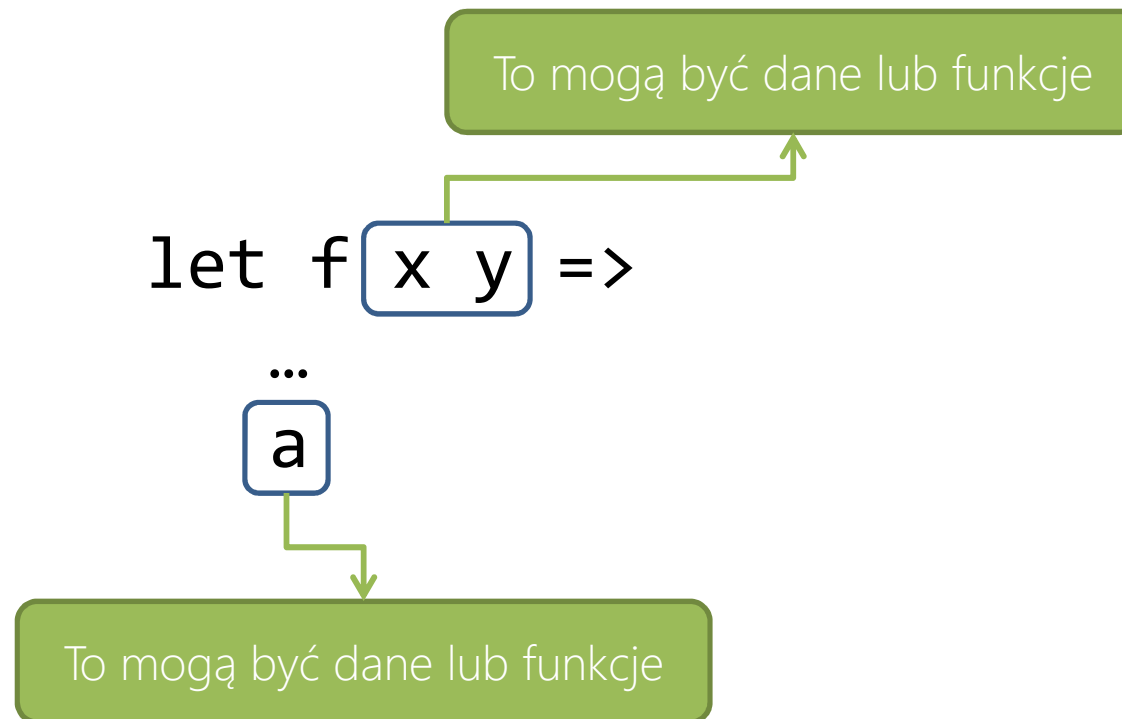


FUNKCJE WYŻSZYCH RZĘDÓW

Funkcje wyższych rzędów



Przykład

```
let rec iloczyn = function
| Pusta -> 1.0
| Element(g,o) -> g*iloczyn o
```

```
let rec suma = function
| Pusta -> 0.0
| Element(g,o) -> g+suma o
```

```
let listaCalk = stworz 10
```

```
let il = iloczyn
      listaCalk;;
```

```
let su = suma
      listaCalk;;
```

Przykład

```
let rec iloczyn = function  
| Pusta -> 1.0  
| Element(g,o) -> g * iloczyn o
```

Wartość początkowa

```
let rec suma = function  
| Pusta -> 0.0  
| Element(g,o) -> g + suma o
```


Działanie

Jakie są elementy
wspólne tych
dwóch funkcji?



Przykład

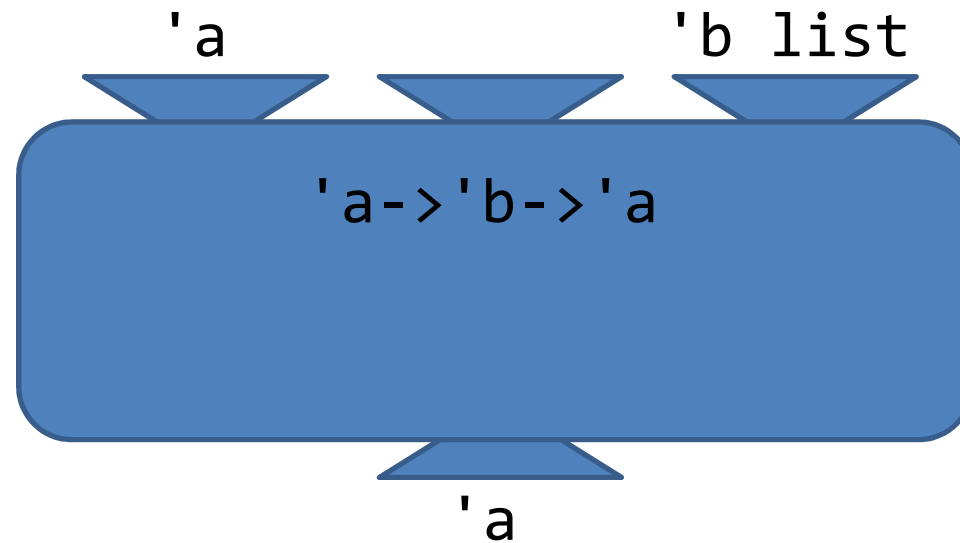
'a 'a->'b->'a



```
let rec agregacja pocz funkcja = function  
| Pusta -> pocz  
| Element (g,o) -> funkcja (agregacja pocz funkcja o) g
```

Przykład

```
let rec agregacja pocz funkcja = function  
| Pusta -> pocz  
| Element (g,o) -> funkcja (agregacja pocz funkcja o) g
```



Przykład

```
let rec agregacja pocz funkcja = function  
| Pusta -> pocz  
| Element (g,o) -> funkcja (agregacja pocz funkcja o) g
```

```
let f1 a x = a+x;;  
let f2 a x = a*x;;  
  
agregacja 0 f1 lista;;  
agregacja 1 f2 lista;;
```

Przykład

```
let rec agregacja pocz funkcja = function  
| Pusta -> pocz  
| Element (g,o) -> funkcja (agregacja pocz funkcja o) g
```

```
let f1 a x = a+x;;  
let f2 a x = a*x;;  
  
agregacja 0 f1 lista;;  
agregacja 1 f2 lista;;
```

```
agregacja 0 (+) lista;;  
agregacja 1 (*) lista;;
```

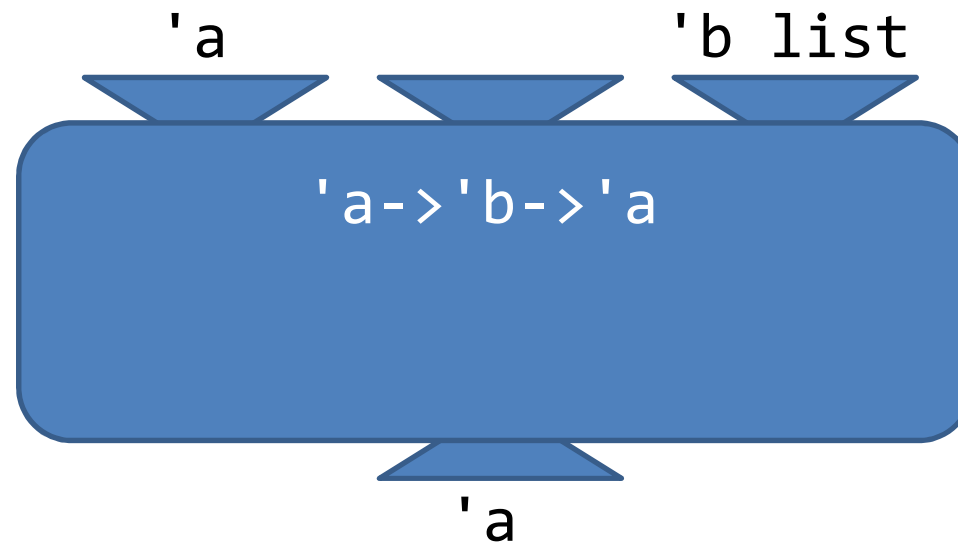

Przykład

```
let rec agregacja pocz funkcja = function  
| Pusta -> pocz  
| Element (g,o) -> funkcja (agregacja pocz funkcja o) g
```

```
agregacja 0 (fun a x -> if x%2 = 0 then a+x else a) lista  
agregacja 0 (fun a x -> if x%3 = 0 then a+x else a) lista
```

Przykład

```
let f1 a x = if x%2 = 0 then a+x else a  
let f2 a x = if x%3 = 0 then a+x else a  
  
let f3 a x v = if x%v = 0 then a+x else a
```





WE NEED TO
MAKE THIS...

...FIT INTO A
HOLE MADE
FOR THIS...

...USING ONLY THIS!

Przykład

'a -> 'b -> 'a



```
let f v = fun a x -> if x%v = 0 then a+x else a
```

```
agregacja 0 (f 2) lista
```

```
agregacja 0 (f 3) lista
```

```
let f3 a x v = if x%v = 0 then a+x else a
```

```
let f v = fun a x -> f3 a x v
```