POPL (Ass-3)

Kousshik Raj
17CS30022

1) Func1 a b c (Func 2 5)

Func1 first is called with a, then b, then c, which finally returns a func (say, func3).

= func3 (Func 2 5)

↳ This is evaluated only when the value is needed. If its not needed, this expn. isn't evaluated.

If its needed, Func takes 2, then 5, as parameters & returns a value which is used by func3 & is evaluated.

Haskell uses currying & lazy propagati

2) In the list comprehension (infinite) sum of the values of pairs is always k.

∴ it is the 12th index (0-based index)

3 a) [1,2,2,4,3,6,4,8,5,10]

b) ["tac", "god", "ant", "nep"]

c) [[1,7] [1,3,5,3,15], [5,7], [1,1,3,1,3,3]]

d) True

4. insert Element : (Eq a, Num b, Eq b).

" " a (c:xs) b =

" " a c 0 = a

" " a [c] 1

5) input data char
extractnonuppercase : string → string
" " "
a = title isUpper a

Kausshik Roy
17CS30022

6 a) b
b) 54
c) (quote cons)
d) ((abc) (aa) (a))

7.
```
(define sum (
    lambda (ls) (
        apply + (cons '0 ls)
    )))
```

8. a)
```
(define counter(
    lambda (ls)(
        if (null? ls) 0
        (+ 1 (sum (cdr ls)))
    )))
```

b)
```
(define AND (lambda (a, b)
    (if (not a) #f
        (if (not b). #f #f)
    )))
```

9.
```
(define multn (ls n) ( mapcar(
    lambda(x)(* n x)) ls ))
```

10.
```
(define Power (x, y)
    ( cond ((= y 0) 1)
        (+ (* x ( power x (- y 1 ))))))))
```