6.2

```
;Aufgabe 6.2
;a
(check-expect (evalQuote '(1 2)) '(1 2))
(check-expect (evalQuote '((+ 1 2) 2)) '(3 2))
(check-expect (evalQuote '(1 (+ 2 2) 2)) '(1 4 2))
(check-expect (evalQuote '((1 2) (/ 2 2))) '((1 2) 1))

(define (evalQuote l)
  (cond [(empty? l) empty]
        [(number? (first l))(cons (first l) (evalQuote (rest l)))]
        [(symbol? (first l))((symbol->func (first l)) (second l)(third l))]
        [(cons? (first l))(cons (evalQuote (first l))(evalQuote (rest l)))]))

(define (symbol->func op)
  (cond [(symbol=? op '+) +]
        [(symbol=? op '-) -]
        [(symbol=? op '*) *]
        [(symbol=? op '/) /]))

;b
(define (zeile name rot gruen blau)
`(tr (td, name)
     (td, rot)
     (td, gruen)
     (td, blau)
     (td, (+ (* 0.299 rot) (* 0.587 gruen) (* 0.114 blau)))))
```