```haskell
import System.IO


data Line = Line {ln::Int, txt::String}
    deriving (Eq,Show)


type Lines = [Line]


evenLines :: Lines -> Lines
evenLines = filter (\l ->  ln l `mod` 2 == 0)

mkLines :: Int -> [String] -> Lines
mkLines _ []= []
mkLines n (l:ls) =
    if null l then mkLines (n+1) ls else Line n l : mkLines (n+1) ls


dropSp :: String -> String
dropSp s = reverse $ dropWhile (==' ') $ reverse s

readLnFile :: String -> IO ()
readLnFile fname = do
    h <- openFile fname ReadMode
    c <- hGetContents h
    putStrLn $ unlines $ map txt $ evenLines $ mkLines 1 $ lines c
    hClose h



{-

LET T = \ x y . x y
LET F = \ x y . y


LET NOR = \ a b . a (\ t . F) (b (\ t . F) T)

NOR T F = (\ a b . a (\ t . F) (b (\ t . F) T)) T F =
        = (\ b . T (\ t . F) (b (\ t . F) T)) F =
        = T (\ t . F) (F (\ t . F) T) =
        = (\ x y . x y) (\ t . F) (F (\ t . F) T) =(2beta)
        = (\ t . F) (F (\ t . F) T) =
        = F

-}


data Time = T {hh::Int, mm::Int, ss::Int}
    deriving (Eq, Show)

type Ticks = [Time]


invTime :: Time -> Bool
invTime (T h m s) =
    h>23 || m>59 || s>59

wrongTime :: Ticks -> Ticks
wrongTime = filter invTime

parseLine :: String -> Time
parseLine l = T (read h) (read m) (read s)
    where
        h = take 2 l
        m = take 2 $ drop 3 l
        s = drop 6 l

t2s :: Time -> String
t2s (T h m s) =
    sh h ++":"++ sh m ++":"++ sh s
    where
        sh n =
```

```
            let sv = (show :: Int -> String) n
            in if length sv < 2 then '0':sv else sv

readF fname = do
    h <- openFile fname ReadMode
    c <- hGetContents h
    putStrLn $ unlines $ map t2s $ wrongTime $ map parseLine $ lines c
    hClose h


{-

LET T = \ x y . x
LET F = \ x y . y x


LET NAND = \ a b . a (b F (\ f . T)) (\ f . T)

NAND F T = (\ a b . a (b F (\ f . T)) (\ f . T)) F T =
         = (\ b . F (b F (\ f . T)) (\ f . T)) T =
         = F (T F (\ f . T)) (\ f . T) =
         = (\ x y . y x) (T F (\ f . T)) (\ f . T) =(2beta)
         = (\ f . T) (T F (\ f . T)) =
         = T

-}
```