Lab Exercise 3

1) Write the following functions using list comprehension.

a) nestedOdds :: [[Int]] -> [[Int]] that takes a list of list of integers and returns it after removing all even integers. (Hint : Use nested list comprehension)

```
>>nestedOdds [[1,2],[4,6],[6,7,8,9]]
[[1],[],[7,9]]
```

b) altMap :: (a -> b) -> (a -> b) -> [a] -> [b] - that takes two functions and a list and returns the list after applying the two functions alternatively to list elements.

```
>>altMap (+1) (+2) [1..5]
[2,4,4,6,6]
```

2) Write the following functions using foldr and foldl.

a) length - returns the length of a list
b) (++) - append two lists
c) product - computes the product of a list of numbers
d) or - returns the disjunction of a boolean list
e) any - apples to a predicate and a list and returns True if any element of the list satisfies the predicate
f) all - applies to a predicate and a list - determines if all the elements satisfy the predicate
g) map
h) reverse
i) concat - concatenate a list of list into single list
j) elem
k) filter
l) partition - takes a predicate and a list and returns the pair of list of elements that do and do not satisfy the predicate

```
>>partition even [1,4,2,3,5,6]
([4,2,6],[1,3,5])
```

m) unzip :: [(a,b)] -> ([a],[b]) - transforms a list of pairs into a list of first component and list of second component

n) intersperse - takes an element and a list and “intersperses” that element between the elements of the list.

```
>>intersperse ‘,’ “abcd”
“a,b,c,d”
```

o) takeWhile - applies to a predicate p and a list xs,returns the longest prefix (possibly empty) of xs of elements that satisfy p

p) tails :: [a] -> [[a]] - returns all suffixes (or final segments) of the list, longest first.

```
>>tails “abc”
[“abc”,”bc”, “c”, “ “]
```