1. map map 为函数，类型为 (?.X1 -> ?.X2) list -> (?.X1 list -> ?.X2 list) list,
2. 结果为：[[2,4],[6,8,10],[12,14,16,18]]
3. 结果为:[[],["four","five"],["year"]]
4. nrev: Work(n) = 1+ Work(n-1) = 1 +1 + Work(n-1) ,递推易知Work(nrev) = O(n)

revAppend: Work(revAppend) = O(n)

区别：revAppend需要而外开辟一个内存空间来存储结果。

3 功能：take，截取数组的前n个数，顺序不变

rtake, 截取数组的前n个数，同时将其逆序放入token前，相当于reverse(take(L,i))@taken

性能：take的Work为O(n)

rtake的Work为O(n)

4 .  fun treeFilter f a  *=*  *case* (f a) *of*

   false *=>* NONE

*|* true *=>* SOME  a;

5.

fun append(xs, ys) =

if null xs

then ys

else (hd xs) :: append(tl xs, ys)

fun splitAt(xs *:* int list, x : int) =

let

    fun partition(ys *:* int list, more : int list, less : int list) =

        if null ys

        then (more, less)

        else

            if hd ys < x

            then partition(tl ys, more, append(less, [hd ys]))

            else partition(tl ys, append(more, [hd ys]), less)

in

    partition(xs, [], [])

end

fun qsort(xs *:* int list) =

  if length xs <= 1

  then xs

  else

    let

        val s = splitAt(xs, hd xs)

    in

        qsort(append(#2 s, #1 s))

    end

6.

fun trav(Empty) *=* [[]]

| trav(Node(l,x,r)) =

*let*

    val left *=*  trav(l);

    val right *=*  trav(r);

    val lr *=* map (fn R *=>* 0::R) left;

    val rr *=* map (fn R *=>* 1::R) right;

*in* [[]]@lr@rr

*end* ;