作业2

1.请用归纳法证明isort()函数的正确性（通过ins()函数来证明）

首先列出ins()函数与isort()函数：

```
fun ins x [] = [x]
    | ins x (y::ys) =
        if x < y then x::y::ys
        else y :: (ins x ys)

fun isort [] = []
    | isort (x::L) = ins x (isort L)
```

ins()的作用是将一个数插入到升序数组中，并保持数组的升序性，其能匹配到两种情况：

1. 当数组为空时，直接插入。
2. 当数组不为空时，对比该数(`x`)与数组第一个元素(`y`)，并得到剩余数组(`ys`)：
   1. x<y时，将 `x, y, ys` 依次连接并返回
   2. `x >= y`时, 将 x 插入到 ys 中

利用ins()编写的isort()函数有以下分析：

1. ins()初始条件(上述第一点)成立
2. 每次迭代会使数组长度减一, 最终一定能终止
3. 迭代条件(上述第二点)能保证数组一定是升序的，进而能够使用ins()进行插入排序

2.分析下列表达式的类型：

```
fun all(your, base)=
      case your of
      0=> base
      | _=> "are belong to us"::all(your-1, base)
```

类型：`fn : int * string list -> string list`
作用：将"are belong to us"重复int(your)次，与base连接后返回

```
fun funny (f, []) = 0

    | funny(f, x::xs)=f(x, funny(f, xs))
```

类型：`fn : (a * int -> int) * a list -> int`
作用：取出第二个参数的每一项，作为f的第一个参数，然后将funny实现reduce的功能，作为第二个参数，进行返回。

```
(fn x => (fn y => x)) "Hello, World!"
```

类型: `fn : ANY -> string`
作用: 该表达式的值是一个函数, 函数接受一个参数, 但是不管传入什么参数都会返回字符串 "Hello, World!"

3.分析下列函数的执行性能

```
fun fib n = if n<=2 then 1 else fib(n-1) + fib(n-2);

fun fibber (0: int) : int * int = (1, 1)
        | fibber (n: int) : int * int =
                let val (x: int, y: int) = fibber (n-1)
                in (y, x + y)
                end
```

1. fib采用递归的方式计算斐波那契数列，但由于重复计算，复杂度为$O(2^n)$
2. fibber采用迭代方式计算斐波那契数列，从前往后推，不会重复计算，复杂度为$O(n)$