标准库提供了几种迭代器，包括迭代文件每行的io.lines，迭代table元素的pairs，迭代数组元素的iparis，迭代字符串单词的string.gmatch等等，lua手册中对pairs与ipairs解释如下：

**ipairs**

return three values: an iterator function, the table t, and 0, so that the construction

for I, v in ipairs(t) do body end

will iterate over the pairs(1, t[1]),(2, t[2])… up to the first integer key absent（缺席） from the table

实际上也就是r[i] == nil的时候就返回了

**paris**

return three values: the next function, the table t, and nil, so that the construction

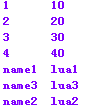
for k, v in pairs(t) do body end

will iterate over all key-value pairs of table t

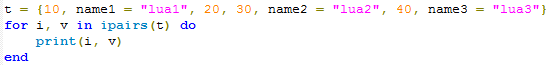
pairs可以遍历表中所有的key，并且除了迭代器本身以及遍历表本身还是返回nil

ipairs则不能返回nil，只能返回数字0，如果遇到nil则退出，它只能遍历到表中出现的第一个不是整数的key





注意lua官方文档中说遍历key-value值的顺序是不确定的，与可能的hash值有关

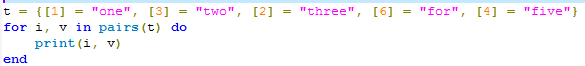




注意如果数组的起始下表不为1呢？

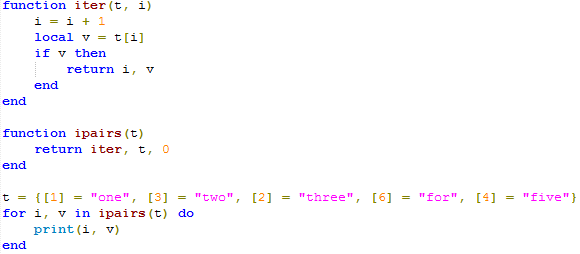


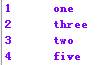






我们也可以实现自己的ipairs





我们也可以实现自己的pairs：

Lua中的next函数：

Allow a program to traverse all fields of a table. Its first argument is a table and its second argument is an index in this table. Next returns the next index of the table and its associated value. When called with nil as its second argument, next returns an initial index and its associated value. When called with the last index, or with nil in an empty table, next returns nil. If the second argument is absent, then it is interpreted ad nil. In particular, you can use next(t) to check whether a table is empty.

The behavior of next is undefined if during the traversal, you assign any value to a non-exitent field in the table. You may however modify existing fields. In particular, you may clear existing fields



