In [1]:
```
a = [
[0,0,0,3],
[0,3,0,4],
[0,0,0,0],
[1,0,0,0]
]                # the sudoku that need to be solved, use zero instead of the
 empty spot

v_all = {1,2,3,4} #the number set
countzeros = (sum(a,[])).count(0) #counts the number of zero in a given sudoku


def showit ():
    for i in range(4):
        print()
        for j in a[i]:
            print (str(j), end=" ")
    print() # create the sudoku

showit()

while countzeros > 0:
    for r in range(4):
        for c in range(4):
            if a[r][c] == 0:
                srow = set (a[r]) # turn the zeros in a row into a set, which
 makes the process of lookingup the element easier
                scol = set ([a[x][c] for x in range(4)]) # turn the zeros in a
column into a set
                br = r - r%2
                bc = c - c%2
                sblk = {a[br][bc], a[br][bc+1], a[br+1][bc], a[br+1][bc+1]}
                # finding the top left element of the small square (2*2)
                s = v_all - srow - scol - sblk
                if len(s) == 1: # if the remaining set has only one element le
ft, we will put this element into the sudoku; if the remaining set has more th
an one element left, we will skip to the next step
                    a[r][c] = next (iter(s))
                    showit()
    if (sum(a,[])).count(0) < countzeros: # turn a into a single dimension and
count the number of zero in this list
        countzeros = (sum(a,[])).count(0)
    else:
        print ("The sudoku can not be solved with this algorithm.")
        break
print ()
```

```
0 0 0 3
0 3 0 4
0 0 0 0
1 0 0 0

0 0 0 3
2 3 0 4
0 0 0 0
1 0 0 0

0 0 0 3
2 3 1 4
0 0 0 0
1 0 0 0

0 0 0 3
2 3 1 4
0 0 0 0
1 0 0 2

4 0 0 3
2 3 1 4
0 0 0 0
1 0 0 2

4 1 0 3
2 3 1 4
0 0 0 0
1 0 0 2

4 1 2 3
2 3 1 4
0 0 0 0
1 0 0 2

4 1 2 3
2 3 1 4
3 0 0 0
1 0 0 2

4 1 2 3
2 3 1 4
3 0 4 0
1 0 0 2

4 1 2 3
2 3 1 4
3 0 4 1
1 0 0 2

4 1 2 3
2 3 1 4
3 0 4 1
1 4 0 2

4 1 2 3
2 3 1 4
```

```
3 0 4 1
1 4 3 2

4 1 2 3
2 3 1 4
3 2 4 1
1 4 3 2
```

In [ ]: