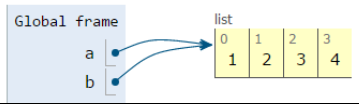
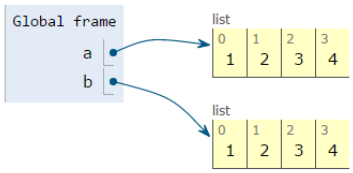
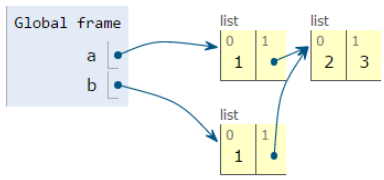
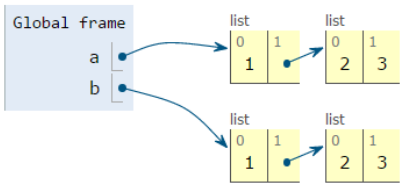
2559_2_Recursive_L4_CopyList

โจทย์ข้อนี้ยาวหน่อย แต่เขียนฟังก์ชันสั้นมาก ๆ

หากเราเขียน **a = [1,2,3,4]; b = a** **a** กับ **b** จะอ้างอิง **list** เดียวกัน เมื่อทำ **a[0] = 9** **b[0]** ก็เปลี่ยนเป็น **9** ด้วย	
ถ้าต้องการเป็นคนละ **list** ก็ต้องใช้ **b = list(a)** แบบนี้ **a[0] = 9** **b[0]** มีค่าเดิมคือ **1**	
แต่ก็อาจเป็นปัญหากับบางงาน เพราะเขียน **b = list(a)** จะทำงานเหมือนคำสั่ง **b = list()** **for e in a: b.append(e)** เป็นการนำ **e** ไปเก็บใน **b** ปัญหาคือถ้ามี **e** ใน **a** ที่เป็น **list** (นั่นคือ **a** เป็น **list** ที่ภายในมี **list**) ก็เกิดปัญหาอีก เช่น **a = [1,[2,3]]** **b = list(a)** **a[0]** กับ **b[0]** ไม่กระทบกัน แต่ **a[1]** กับ **b[1]** ก็ยังอ้างอิง **list [2,3]** เดียวกัน แปลว่า เปลี่ยน **a[1][0] = 99** **b[1][0]** ก็เปลี่ยนด้วย	
ถ้าจะแก้ปัญหาข้างบนนี้ การทำงานก็ต้องเป็น **a = [1,[2,3]]** **b = list(a)** **b[1] = list(a[1])** ในกรณีทั่วไป **a** เป็น **list** ซ้อนด้วย **list** ก็ต้องตรวจแต่ละช่องใน **a** ว่าเป็น **list** หรือเปล่า ถ้าไม่ใช่ก็แล้วไป ถ้าใช่ก็ต้อง **copy** ต่อ ถ้า **a** เป็น **list** ซ้อน **list 2** ชั้น เช่น **[1,2,[3,4],[5,6],8]** ก็เขียนแบบนี้ **a = [1,2,[3,4],[5,6],8]** **b = list()** **for e in a:** **if type(e) is list:** **b.append(list(e))** **else:** **b.append(e)** แต่ถ้า **a** เป็น **list** ซ้อนด้วย **list** ซ้อนด้วย **list ...** กี่ชั้นก็ได้ ก็จะเหมาะมากกับการเขียนโปรแกรมแบบ **recursive**	

จึงเป็นที่มาของโจทย์นี้ ที่ต้องการฟังก์ชันเพื่อ **copy list** ที่เป็น **list** ซ้อนด้วย **list** ซ้อนด้วย **list ...** กี่ชั้นก็ได้

จงเขียนฟังก์ชัน **copylist(a)** ของโครงโปรแกรมข้างล่างนี้

copylist รับ **a** เป็น **list** เก็บจำนวนเต็มหรือ **list**

แล้วคืน **list** อีกตัวที่มีข้อมูลภายในเหมือน **a** ทุกประการ แต่เป็นการสร้างใหม่หมด

โดย **a** เป็น **list** ที่ภายในก็เก็บ **list** ซ้อน **list** กี่ชั้นก็ได้ ดังตัวอย่างที่แสดงข้างล่างนี้

```
def copylist( x ):

  ???


exec(input().strip())  # do not remove this line
```

ข้อมูลนำเข้า

คำสั่งในการทดสอบฟังก์ชันที่เขียน

ข้อมูลส่งออก

ผลที่ได้จากคำสั่งที่ป้อนเป็นข้อมูลนำเข้า

ตัวอย่าง

input	Output
`print(copylist([[[[],[],[]]]]))`	`[[[[], [], []]]]`
`a=[[[1]]];b=copylist(a);print(b,a==b,id(a[0])!=id(b[0]))`	`[[[1]]] True True`
`a=[1,[2,[3]]];b=copylist(a);print(b,a==b, id(a[1][1])!=id(b[1][1]))`	`[1, [2, [3]]] True True`

ความรู้เพิ่มเติม:

คำถาม: เราจะรู้ได้อย่างไรว่า **list** สอง **lists** ที่มีค่าเหมือนกัน เป็นตัวเดียวกันหรือเปล่า **?**

คำตอบ: ใช้ฟังก์ชัน **id** ของ **python** ที่จะคืนหมายเลขประจำตัว ถ้าหมายเลขเดียวกันก็แสดงว่า เป็นที่เก็บเดียวกัน เช่น

```
a = [1,2]
b = a
c = list(a)
print(id(a),id(b),id(c))
```

จะพบว่า **id(a)** เหมือน **id(b)** แต่จะต่างกับ **id(c)**