Assignment 3: Treasure Hunt

เกมล่าขุมทรัพย์

โลกขุมทรัพย์เป็นตารางขนาด N*N

คำสั่ง เป็นสายอักขระ ประกอบด้วยตัวอักษร F L R เช่น "FFLFFR"

การหาขุมทรัพย์ให้เดินตามคำสั่ง

เมื่อเจอ F ขยับไปข้างหน้าหนึ่งช่อง

เมื่อเจอ L ให้หมุนซ้าย 90 องศา

เมื่อเจอ R ให้หมุนขวา 90 องศา

โปรแกรมที่ให้ไว้ประกอบด้วย

กำหนดจุดเริ่มต้น origin(x,y)

เดินตามคำสั่งในแผนที่ makemove(m)

พิมพ์โลกขุมทรัพย์ออกมาดู printworld()

จากตัวอย่างฟังก์ชัน test() คำสั่งคือ "FFRFFLFFF"

```
def test():
    origin(1,1)
    mymap = "FFRFFLFFF"
    makemove(mymap)
    printworld()
```

โลกขนาด 10*10 กำหนดให้ ตำแหน่ง (0,0) เป็นมุมล่างซ้าย เริ่มต้นหันหน้าไปทางทิศเหนือ (หัวตั้งขึ้น)

เมื่อ run ฟังก์ชัน test() จะพิมพ์ทางเดินในโลกขุมทรัพย์ออกมาดังนี้

```
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0
0 1 1 1 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

งานของคุณคือ

คำสั่งถูกฉีกเป็นหลายส่วน คุณมีหน้าที่เขียนฟังก์ชันให้ประกอบคำสั่งเหล่านี้เข้าด้วยกัน

สมมติก่อนว่าความยาวคำสั่งเป็นเลขคู่

มีงานบังคับให้ทำสามแบบ

1 คำสั่งถูกแยกเป็นสองชิ้น จับมันมาต่อกัน

2 คำสั่งมีสองชิ้น แต่ละชิ้นถูกแบ่งครึ่ง การต่อเข้าด้วยกัน ให้สลับชิ้นส่วนกับแผ่นอื่น เช่นชิ้นหนึ่งเป็น "FFFF" ชิ้นสองเป็น "RRLL" เมื่อประกอบแล้วเป็น "FFRRFFLL"

3 สุดท้ายยากขึ้น คำสั่งมีสองชิ้น เวลามาต่อกันให้สลับอักขระสองชิ้นเข้าด้วยกันทีละตัว เช่น "FFFF" และ "RLRL" ต่อกันเป็น "FRFLFRFL"

โดยคุณเขียนฟังก์ชัน joinmap(s1,s2) เพื่อรวม คำสั่ง s1 s2 เข้าด้วยกัน

```
def joinmap(s1,s2):
    #   เติมโปรแกรมของคุณที่นี่
    return s
```

และ run จาก main()

```
def main():
  origin(1,1)
  s1 = "FFLL"
  s2 = "RRFFFF"
  s = joinmap(s1,s2)
  makemove(s)
  printworld()
```

กำหนด s1 s2 เอาเอง เมื่อโปรแกรมทำงานเสร็จ ดูด้วยตาว่า การเดินถูกต้องตามที่คุณคาดหรือไม่

นอกจากนี้

คุณอาจคิดถึงการใส่รหัสลับในคำสั่ง แล้วมีการแปลรหัสก่อนเป็นต้น

หรือคุณสามารถเพิ่มฟังก์ชัน เช่นในโลกขุมทรัพย์มีสิ่งกีดขวางเมื่อคำสั่งเดินไปชนสิ่งกีดขวางจะเดินไปต่อไม่ได้ เป็นต้น เมื่อคุณทำเพิ่มเติมจากที่บังคับสามข้อ จะมีคะแนนพิเศษให้

โค้ดเริ่มต้นที่ให้

```
N = 10    # size of world N * N
world = [[0]*N for i in range(N)]

# bottomleft is (0,0), x goes right, y goes up
def printworld():
    for i in range(N-1,-1,-1):
        for j in range(N):
            print(world[j][i], end = " ")
        print()
```

```
hd = 0   # heading 0-N, 1-E, 2-S, 3-W
px = 0   # current position x
py = 0   # current position y

# move forward one step
def forward():
    global hd, px, py
    # move one step
    if(hd == 0):
        py += 1
    elif(hd == 1):
        px += 1
    elif(hd == 2):
        py -= 1
    elif(hd == 3):
        px -= 1
    # constrain x,y in bound 0..N-1
    if(px > N-1):
        px = N-1
    if(px < 0):
        px = 0
    if(py > N-1):
        py = N-1
    if(py < 0):
        py = 0
    world[px][py] = 1

# turn head left 90 degree
def turnleft():
    global hd
    hd -= 1
    if(hd < 0):
        hd = 3

# turn head right 90 degree
def turnright():
    global hd
    hd = (hd + 1) % 4

# make move according to m (map)
def makemove(m):
    for c in m:
        if(c == "F"):
            forward()
        elif(c == "L"):
            turnleft()
        elif(c == "R"):
            turnright()

def origin(x,y):
    global px,py
    px = x
    py = y
    world[px][py] = 1
```

```
def test():
    origin(1,1)
    mymap = "FFRFFLFFF"
    makemove(mymap)
    printworld()

def joinmap(s1,s2):
    # put your code here
    return s

def main():
    # test()
    origin(1,1)
    s1 = “FFFLLL”
    s2 = “FFRRFF”
    s = joinmap(s1,s2)
    makemove(s)
    printworld()

main()

-------------------------end ------------------
```