PRACTICE CERTIFICATION COMPETE JOBS LEADERBOARD Search hackerrank682

Practice > Algorithms > Implementation > Matrix Layer Rotation > Editorial

Matrix Layer Rotation ☆

897.7 more points to get your next star!

Rank: 75298 | Points: 1302.3/2200

Problem Submissions Leaderboard Discussions Editorial

Editorial by abhiranjan

Suppose we have following 4x5 matrix.

```
a11  a12  a13  a14  a15

a21  a22  a23  a24  a25

a31  a32  a33  a34  a35

a41  a42  a43  a44  a45
```

In order to rotate it, we create strips of various levels. In the above example 3 strips will be created and they are as follows.

```
Outer strip:

a11  a12  a13  a14  a15

a21                 a25

a31                 a35

a41  a42  a43  a44  a45

->

a11 <- a12 <- a13 <- a14 <- a15 <- a25 <- a35 <- a45 <- a44 <- a43 <- a42 <- a41 <- a31 <- a2

Inner strip:

     a22  a23  a24

     a32  a33  a34

->

a22 <- a23 <- a24 <- a34 <- a33 <- a32
```

Since R can be very large, we will rotate each strip, of length k, by R%k times. After rotating each strips we will recreate the original matrix.

Let's rotate above strip by one step, R = 1, and the creating the matrix will involve the following steps.

```
Outer strip:

a12 <- a13 <- a14 <- a15 <- a25 <- a35 <- a45 <- a44 <- a43 <- a42 <- a41 <- a31 <- a21 <- a1

->

a12  a13  a14  a15  a25

a11                 a35

a21                 a45

a31  a41  a42  a43  a44


Inner strip:

a23 <- a24 <- a34 <- a33 <- a32 <- a22

->

     a23  a24  a34
```

STATISTICS

Difficulty:	Hard
Time Complexity:	$O(N \times M)$
Required Knowledge:	Implementation
Publish Date:	Jun 29 2015

This is a Practice Challenge

NEED HELP?

View discussions

View top submissions

```
    a22  a32  a33
```

So the final configuration of the above matrix after 1 rotation will be:

```
a12  a13  a14  a15  a25

a11  a23  a24  a34  a35

a21  a22  a32  a33  a45

a31  a41  a42  a43  a44
```

Set by gman007

Problem Setter's code:

Scala

```
import scala.collection.mutable.HashMap;

object Solution{
    def main(args: Array[String]) = {
        val in = readLine.split(" ").map(_.toInt);
        var arr = new Array[Array[Int]](in(0));
        for(i <- 1 to in(0))
            arr(i - 1) = readLine.split(" ").map(_.toInt);
        println(rotate(arr, in(2)));
    }

    def rotate(arr:Array[Array[Int]], r:Int):String = {
        var b = Array.fill(arr.size){ new Array[Int](arr(0).size) };
        var m = arr.size;
        var n = arr(0).size;
        val min = Math.min(m/2, n/2);
        for(i <- 1 to min){
            val map = getMap(m, n);
            val len = map.keys.size;
            for(j <- map.keys){
                val cur = map(j);
                val next = map((j + r) % len);
                b(next._1 + i - 1)(next._2 + i - 1) = arr(cur._1 + i - 1)(cur._2 + i - 1);
            }
            m -= 2;
            n -= 2;
        }
        return b.map(x => x.mkString(" ")).mkString("\n");
    }

    def getMap(m:Int, n:Int):HashMap[Int, Tuple2[Int, Int]] = {
        var map = new HashMap[Int, Tuple2[Int, Int]]();
        val len = 2 * (m + n - 2) - 1;
        for(i <- 0 to len) map += (i -> getLoc(i, m, n));
        return map;
    }

    def getLoc(i:Int, m:Int, n:Int):Tuple2[Int, Int] = {
        if(i < m - 1)
            return (i, 0);
        else if(i < m + n - 2)
            return (m - 1, (i - m + 1) % n);
        else if(i < 2 * m + n - 3)
            return (2 * m + n - 3 - i, n - 1);
        else
            return (0, 2 * (m + n - 2) - i);
    }
}
```

Tested by shashank21j

Problem Tester's code:

Python 2

```
from copy import deepcopy
```

```
m, n, r = map(int, raw_input().split())
matrix = []
for i in xrange(m):
    matrix.append(map(int, raw_input().split()))
k = min(m, n) / 2
rows = []
for ii in xrange(k):
    row = []
    for i in xrange(ii, m - 1 - ii):
        row.append(matrix[i][ii])
    for i in xrange(ii, n - 1 - ii):
        row.append(matrix[m - 1 - ii][i])
    for i in xrange(m - 1 - ii, ii, -1):
        row.append(matrix[i][n - 1 - ii])
    for i in xrange(n - 1 - ii, ii, -1):
        row.append(matrix[ii][i])
    rows.append(row)

result = deepcopy(matrix)

for ii in xrange(k):
    row = rows[ii]
    shift = r % len(row)
    idx = -shift
    for i in xrange(ii, m - 1 - ii):
        result[i][ii] = row[idx]
        idx += 1
        idx %= len(row)
    for i in xrange(ii, n - 1 - ii):
        result[m - 1 - ii][i] = row[idx]
        idx += 1
        idx %= len(row)
    for i in xrange(m - 1 - ii, ii, -1):
        result[i][n - 1 - ii] = row[idx]
        idx += 1
        idx %= len(row)
    for i in xrange(n - 1 - ii, ii, -1):
        result[ii][i] = row[idx]
        idx += 1
        idx %= len(row)
for i in result:
    print " ".join(map(str, i))
```

Feedback

Was this editorial helpful?

Yes No

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature