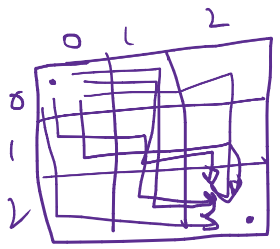
21-Dec-2021

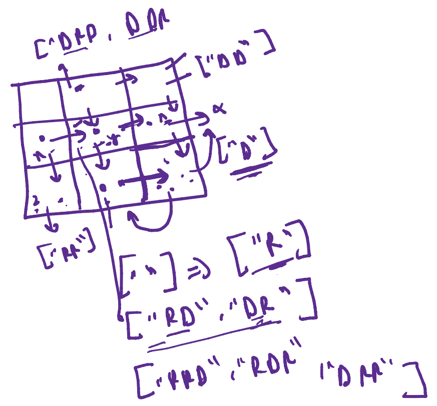
21 December 2021 20:58

getMazePath :- 6

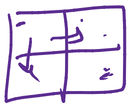


[ "HHVV",
"VVHH" ]

R D



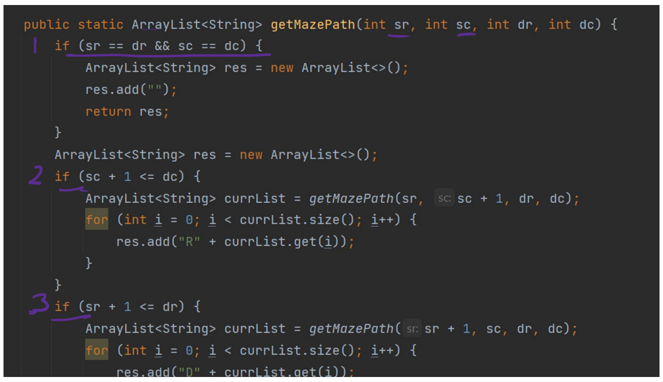
```
AL<String> gMp(sr, sc, dr, dc) {
if(sr==dr && sc==dc) return List[""];
res = new AL();
    if(sc+1 ≤ dc) {
    currlist = gmp(sr, sc+1, dr, dc);
    for(i=0 to currlist.size-1) {
    res.add("R" + currlist.get(i))
    }
}

    } if (sr+1 ≤ dr) {
    currlist = gmp(sr+1, sc, dr, dc)
    for(i=0 to currlist.size-1) {
    res.add("D" + currlist.get(i))
    }
    }
} return res;
}
```



1

dc=1 n+1 ≤ dc

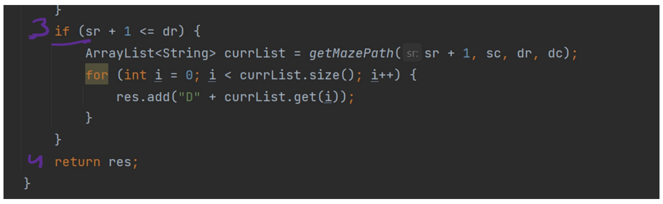
sr sc res



```
public static ArrayList<String> getMazePath(int sr, int sc, int dr, int dc) {
1   if (sr == dr && sc == dc) {
        ArrayList<String> res = new ArrayList<>();
        res.add("");
        return res;
    }
    ArrayList<String> res = new ArrayList<>();
2   if (sc + 1 <= dc) {
        ArrayList<String> currList = getMazePath(sr, sc + 1, dr, dc);
        for (int i = 0; i < currList.size(); i++) {
            res.add("R" + currList.get(i));
        }
    }
3   if (sr + 1 <= dr) {
        ArrayList<String> currList = getMazePath(sr + 1, sc, dr, dc);
        for (int i = 0; i < currList.size(); i++) {
            res.add("D" + currList.get(i));
```

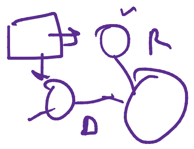
| | | |
|---|---|---|
| 0 | 0 | [RRDD, RDRD, RDDR |

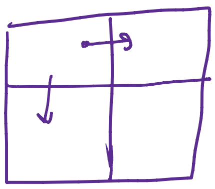
+ ~~2~~ 3

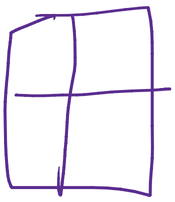


```
    }
3   if (sr + 1 <= dr) {
        ArrayList<String> currList = getMazePath( sr: sr + 1, sc, dr, dc);
        for (int i = 0; i < currList.size(); i++) {
            res.add("D" + currList.get(i));
        }
    }
4   return res;
}
```

dr = 2
dc = 2







$O(2^n)$

$2^k = n$

$k = \log_2(n)$

HashSet

HashMap / HashTable

| Unique Key | Value |
|---|---|
| — | ________ |
| — | ________ |

get = $O(1)$

add = $O(1)$

$= 2n = O(n)$

put = (key, value)

get = (key)

remove = (key)

| K | V |
|---|---|
| 1 | A |
| 2 | Z |
| 3 | C |

2

Student {

no, name

}

$2n = O(n)$

55, Krish

0

Count frequencies :-

i = 0

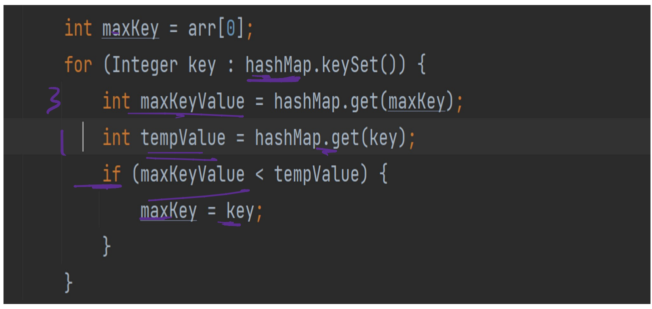
arr = [ 2, 4, 1, 3, 1, 5, 6, 7, 8, 1 ]

i

mk = ~~2~~ 1

HM

| K = Integer | V = Integer |
|---|---|
| 2 | 1 |
| 4 | 1 |
| 1 | 3 |
| 3 | 1 |
| 5 | 1 |
| 6 | 1 |
| 7 | 1 |
| 8 | 1 |



```
int maxKey = arr[0];
for (Integer key : hashMap.keySet()) {
    int maxKeyValue = hashMap.get(maxKey);
    int tempValue = hashMap.get(key);
    if (maxKeyValue < tempValue) {
        maxKey = key;
    }
}
```