题解

目录

#A、动态联通块

题目大意

一个 $n \times m$ 的方格图

一些格子被涂成了黑色,在方格图中被标为 1 ,白色格子标为 0

一个联通块是指这个联通块内全是 0 格子,且上下左右四个方向联通

现在对每个位置的 1 ,假设它变为 0 后,则他所在的联通块的大小是?

对于矩阵的第 i 行第 j 列

如果原来的方格图上是 0 ,则输出这个位置在原来的方格图上所属的联通块大小

否则输出将初始方格图上仅将这个方格变为 0 后新的方格图上,该位置所在的联通块大小

#A、动态联通块

思路

可以先求出每个连通块的大小，并标上编号。对于矩阵的第 i 行第 j 列，如果原来的方格图上是0,则输出这个位置在原来的方格图上所属的联通块大小。否则，可以将这个位置的上、下、左、右在原来的方格图上所属的联通块编号用set去重，求和，然后输出。

时间复杂度是O(n*m)

#C、还原文件

题目大意

一份重要文件被撕成两半,其中一半还被送进了碎纸机

我们将碎纸机里找到的纸条进行编号,如图 1 所示

然后根据断口的折线形状跟没有切碎的半张纸进行匹配,最后还原成图 2 的样子

要求你输出还原后纸条的正确拼接顺序

#C、还原文件

思路

因为数据较为随机且存在梯度，所以可以全排列+剪枝。在枚举第dep层时，如果与断口不匹配，就不用继续枚举了。

#C、还原文件

核心代码

```
bool check(int idx, int cur)
{
  for (int i = 0; i < t[cur].size(); i++)
    if (a[idx + i] != t[cur][i])
      return false;
  return true;
}
```

```
void dfs(int idx, int cur)
{
  if (cur > m)
  {
    for (int i = 1; i < cur; i++)
      printf("%d%c", ans[i], " \n"[i == m]);
    exit(0);
    return;
  }
  for (int i = 1; i <= m; i++)
    if (!vis[i] && check(idx, i))
    {
      vis[i] = true;
      ans[cur] = i;
      dfs(idx + t[i].size() - 1, cur + 1);
      vis[i] = false;
    }
}
```