```
    return 0;
  unordered_map<MemoKey, int, MemoKeyHash> mZ;
  int zerosC = zeros(x, y, !s, m, mZ);
  int minZ = (minZeros != 0)
                ? min(minZeros, zerosC)
                : zerosC;
  MemoKey key = {x, y, minZ, s};
  if (memo.find(key) != memo.end())
  {
    return memo[key];
  }
  int maxA = 0;
  if (minZ == 0)
  {
    maxA = maxArea((s ? x + 1 : x),
                   (s ? y : y + 1), s, 1, minZ, m, memo);
  }
  else
  {
    maxA = max(minZ * k, maxArea((s ? x + 1 : x),
                                 (s ? y : y + 1), s, k + 1, minZ, m, memo));
  }
  memo[key] = maxA;
  return maxA;
}

int max_land(int x, int y, bool s,
             const vector<vector<int>> &m)
{
  int r = 0;
  unordered_map<MemoKey, int, MemoKeyHash> memo;

  if (s && x >= 0 && y >= 0)
  {
    for (int i = x; i < m[y].size(); ++i)
    {
      r = max(r, maxArea(i, y, s, 1, 0, m, memo));
    }
    if (x == 0)
    {
      r = max(r, max_land(x, y - 1, s, m));
    }
    else
    {
      r = max(r, max_land(x - 1, y, !s, m));
    }
  }
  else if (!s && y >= 0 && x >= 0)
  {
    for (int i = y; i < m.size(); ++i)
    {
      r = max(r, maxArea(x, i, s, 1, 0, m, memo));
    }
    if (y == 0)
    {
      r = max(r, max_land(x - 1, y, s, m));
    }
    else
    {
      r = max(r, max_land(x, y - 1, !s, m));
    }
  }
  return r;
}
int main()
{
  int m, n;
  cin >> m >> n;
  while (m != 0 || n != 0)
  {
    vector<vector<int>> grid(m, vector<int>(n));

    for (int i = 0; i < m; ++i)
```