No of Distinct Island in C++

```
#include <iostream>
#include <vector>
#include <unordered_set>

using namespace std;

// Function prototypes
void dfs(vector<vector<int>>& arr, int row, int col,
string& psf);
int numDistinctIslands(vector<vector<int>>& arr);

// Depth-first search to mark all connected land cells of
an island
void dfs(vector<vector<int>>& arr, int row, int col,
string& psf) {
    arr[row][col] = 0; // Marking current cell as visited
    int n = arr.size();
    int m = arr[0].size();

    // Directions: up, right, down, left
    vector<pair<int, int>> dirs = {{-1, 0}, {0, 1}, {1, 0}, {0,
-1}};
    string dirStr = "urdl"; // Corresponding directions
characters

    for (int i = 0; i < 4; ++i) {
        int newRow = row + dirs[i].first;
        int newCol = col + dirs[i].second;
        if (newRow >= 0 && newRow < n && newCol >= 0
&& newCol < m && arr[newRow][newCol] == 1) {
            psf += dirStr[i]; // Append direction character to
path string
            dfs(arr, newRow, newCol, psf);
        }
    }
    psf += "a"; // Append anchor to indicate end of island
path
}

// Function to find number of distinct islands
int numDistinctIslands(vector<vector<int>>& arr) {
    int n = arr.size();
    if (n == 0) return 0;
    int m = arr[0].size();

    unordered_set<string> islands; // Set to store distinct
island paths

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            if (arr[i][j] == 1) {
                string psf = "x"; // Starting character to
represent new island
                dfs(arr, i, j, psf);
                islands.insert(psf); // Insert island path into set
            }
        }
    }

    return islands.size(); // Return the number of distinct
islands
```

Input:

Grid:
1 0 0
0 1 0
1 1 1

Execution Steps

Step 1: Initializing Variables

- The grid has 3 rows x 3 columns.
- An empty set islands is initialized to store distinct island shapes.

Step 2: Traversing the Grid

1. At (0, 0):
    - Start a DFS and encode the path:

      sql
      Copy code
      psf = "x" (start)
      Move down: psf = "xurda" (anchor added after exploring up, right, down, left)

    - Add "xurda" to islands.
2. At (1, 1):
    - Start a DFS and encode the path:

      arduino
      Copy code
      psf = "x" (start)
      No other cells connected to (1, 1): psf = "xurda" (isolated cell)

    - Add "xurda" to islands (already present).
3. At (2, 0):
    - Start a DFS and encode the path:

      sql
      Copy code
      psf = "x" (start)
      Move right: psf = "xrd" (connects (2, 0) → (2, 1))
      Move right again: psf = "xrdrr" (connects (2, 1) → (2, 2))
      Move up: psf = "xrdrru" (connects (2, 2) → (1, 2))
      Add anchors: psf = "xrdrruarrarrarr"

    - Add "xrdrruarrarrarr" to

```
}

int main() {
    // Hardcoded input
    vector<vector<int>> arr = {
        {1, 0, 0},
        {0, 1, 0},
        {1, 1, 1}
    };

    // Calculating number of distinct islands
    cout << numDistinctIslands(arr) << endl;

    return 0;
}
```

islands.

Step 3: Count Distinct Islands

- The set islands contains:

  {"xurda", "xrdrruarrarrarr"}

- The size of the set is 2.

Output:

2

Output:-
2