Goldmine2 in C++

```
#include <iostream>
#include <vector>
using namespace std;

int maxGold = 0;

void travel(vector<vector<int>>& arr, int i, int j,
vector<vector<bool>>& visited, vector<int>& bag) {
    if (i < 0 || j < 0 || i >= arr.size() || j >=
arr[0].size() || arr[i][j] == 0 || visited[i][j]) {
        return;
    }
    visited[i][j] = true;
    bag.push_back(arr[i][j]);
    travel(arr, i - 1, j, visited, bag);
    travel(arr, i, j + 1, visited, bag);
    travel(arr, i, j - 1, visited, bag);
    travel(arr, i + 1, j, visited, bag);
}

void getMaxGold(vector<vector<int>>& arr) {
    int rows = arr.size();
    int cols = arr[0].size();
    vector<vector<bool>> visited(rows,
vector<bool>(cols, false));

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            if (arr[i][j] != 0 && !visited[i][j]) {
                vector<int> bag;
                travel(arr, i, j, visited, bag);

                int sum = 0;
                for (int val : bag) {
                    sum += val;
                }
                if (sum > maxGold) {
                    maxGold = sum;
                }
            }
        }
    }
}

int main() {
    vector<vector<int>> arr = {
        {0, 1, 4, 2, 8, 2},
        {4, 3, 6, 5, 0, 4},
        {1, 2, 4, 1, 4, 6},
        {2, 0, 7, 3, 2, 2},
        {3, 1, 5, 9, 2, 4},
        {2, 7, 0, 8, 5, 1}
    };

    getMaxGold(arr);
    cout << maxGold << endl;

    return 0;
}
```

Sample Grid (Visual):

```
[
{ 0,  1,  4,  2,  8,  2 },
{ 4,  3,  6,  5,  0,  4 },
{ 1,  2,  4,  1,  4,  6 },
{ 2,  0,  7,  3,  2,  2 },
{ 3,  1,  5,  9,  2,  4 },
{ 2,  7,  0,  8,  5,  1 }
]
```

We'll start traversal from **(1,2)** where value = 6

Dry Run Table (DFS Traversal Steps):

Step	Cell Visited	Gold at Cell	Cumulative Sum	Stack (DFS Recursion Path)
1	(1,2)	6	6	(1,2)
2	(0,2)	4	10	(1,2) → (0,2)
3	(0,3)	2	12	(1,2) → (0,2) → (0,3)
4	(0,4)	8	20	...
5	(0,5)	2	22	...
6	(1,5)	4	26	...
7	(2,5)	6	32	...
8	(2,4)	4	36	...
9	(3,4)	2	38	...
10	(3,5)	2	40	...
11	(4,5)	4	44	...
12	(4,4)	2	46	...
13	(4,3)	9	55	...
14	(5,3)	8	63	...
15	(5,4)	5	68	...
16	(5,5)	1	69	...
17	(3,3)	3	72	...
18	(2,3)	1	73	...
19	(2,2)	4	77	...
20	(1,3)	5	82	...
21	(1,1)	3	85	...
22	(2,1)	2	87	...
23	(2,0)	1	88	...
24	(3,0)	2	90	...
25	(4,0)	3	93	...
26	(4,1)	1	94	...
27	(5,1)	7	101	...
28	(5,0)	2	103	...

29	(1,0)	4	107	...
30	(0,1)	1	108	...
31	(3,2)	7	115	...
32	(4,2)	5	120	...

✅ Result:

At the end of this traversal:

- All connected gold cells are visited
- Sum = **120**
- This is the **maximum** among all components

✨ Final Output:

```
Output: 120
```

Output:-

120