

Making A Large Island (/problems/making-a-large-island/)

Submission Detail

75 / 75 test cases passed.

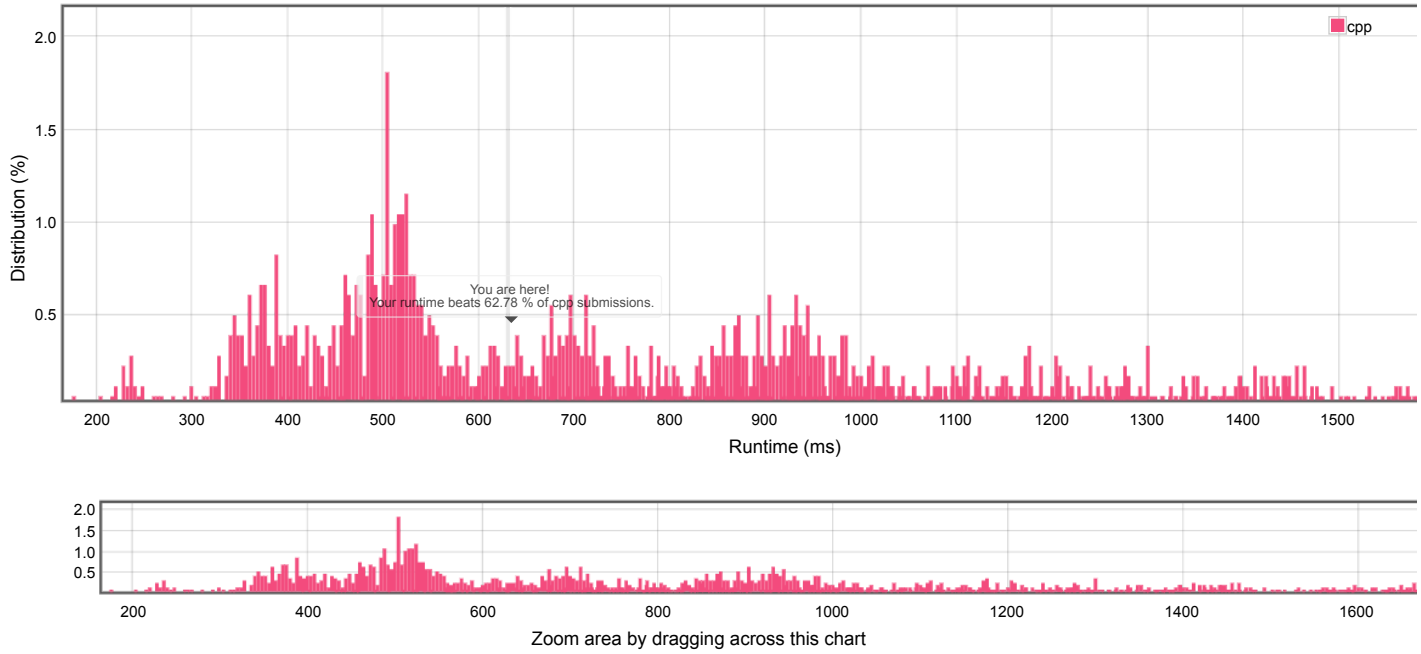
Runtime: **632 ms**

Memory Usage: **171.8 MB**

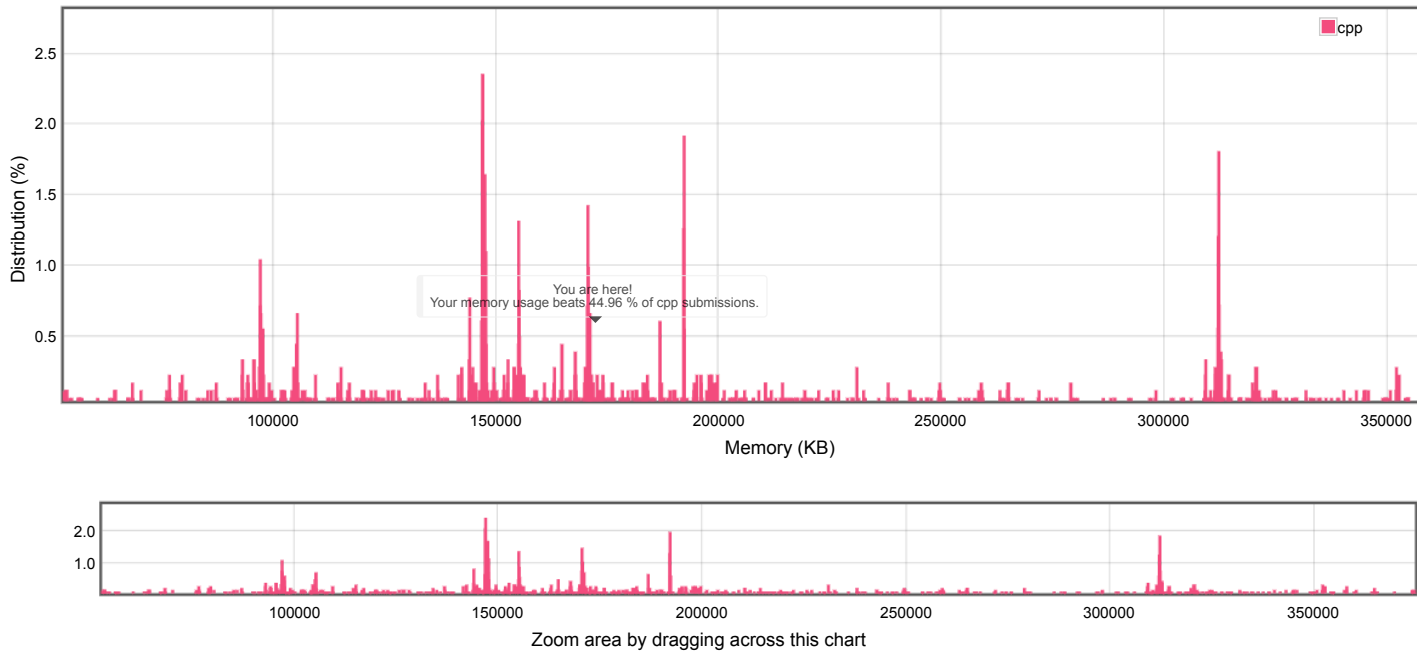
Status: **Accepted**

Submitted: **0 minutes ago**

Accepted Solutions Runtime Distribution



Accepted Solutions Memory Distribution



Invite friends to challenge **Making A Large Island**

Submitted Code: 0 minutes ago

Language: cpp

Edit Code

```
1 class Solution {
2 public:
```

```

3     int n,m;
4     /*almacena el tamaño de cada componente conectado asignando un ID a cada celda del componente conectado que actuará como una clave en el m
5     std::unordered_map<int, int> tam;
6     /*almacena -1 para la celda que no se visito y la identificación de celda para la visitada*/
7     int A[502][502];
8
9
10    // islandNum es el ID
11    int dfs(int i, int j, std::vector<std::vector<int>>& grid, int islandNum){
12        if(i<0 || i>=n || j<0 || j>=m || grid[i][j]==0 || A[i][j]!=-1){
13            return 0;
14        }
15        /*todos los componentes conectados tienen la misma identificación*/
16        A[i][j]=islandNum;
17        int tam=1;
18        tam+=dfs(i+1,j,grid,islandNum);
19        tam+=dfs(i-1,j,grid,islandNum);
20        tam+=dfs(i,j+1,grid,islandNum);
21        tam+=dfs(i,j-1,grid,islandNum);
22        return tam;
23    }
24
25    int largestIsland(std::vector<std::vector<int>>& grid) {
26        n=grid.size();
27        m=grid[0].size();
28        memset(A,-1,sizeof(A));
29        int l=1;
30        for(int i=0;i<n;i++){
31            for(int j=0;j<m;j++){
32                if(grid[i][j]==1 && A[i][j]==-1){
33                    tam[l]=dfs(i, j, grid, l);
34                    l++;
35                }
36            }
37        }
38
39        int maxarea=0;
40        for(int i=0;i<n;i++){
41            for(int j=0;j<m;j++){
42                {
43                    /*si encontramos un 0, intentaremos encontrar los componentes conectados con diferentes IDs*/
44                    if(grid[i][j]==0)
45                    {
46                        int area=1;
47                        std::set<int> s;
48                        if(i+1<n)
49                            s.insert(A[i+1][j]);
50                        if(i-1>=0)
51                            s.insert(A[i-1][j]);
52                        if(j+1<m)
53                            s.insert(A[i][j+1]);
54                        if(j-1>=0)
55                            s.insert(A[i][j-1]);
56                        for(int c: s){
57                            area+=tam[c]; // agregando tam para cada componente conectado único
58                        }
59                        maxarea=std::max(maxarea, area);
60                    }
61                }
62            }
63            if(maxarea!=0)
64                return maxarea;
65            return n*m;
66        }
67    };
68

```

[Back to problem \(/problems/making-a-large-island/\)](/problems/making-a-large-island/)

Copyright © 2021 LeetCode

[Help Center \(/support/\)](/support/) | [Jobs \(/jobs\)](/jobs/) | [Bug Bounty \(/bugbounty\)](/bugbounty/) | [Online Interview \(/interview/\)](/interview/) | [Students \(/student\)](/student/) | [Terms \(/terms\)](/terms/) | [Privacy Policy \(/privacy\)](/privacy/)

 [United States \(/region\)](/region/)