Algorithm 1 Finding Faces

```
1: procedure MAIN()
2:     Input the Graph
3:     c // Vector of Vector of int elements
4:     for each i ∈ Vertex  do
5:         for each j ∈ Vertex do
6:             Initialize(i,j)
7:         end for
8:     end for
9:     WithoutRedundantVectorpaths = RemoveRedundantCircuits(Vectorwithpaths)
10:    FaceMatrix = facesRetrieval(Vectorwithpathsasdataelement)
11: end procedure

12: function INITIALIZE(s, d)
13:    visited[] ← false // array of size V
14:    path[] // vector of size V+1
15:    path_Index gets 0
16:    Circutis(s,u,d,visited,path,path_Index)
17: end function

18: function CIRCUITS(s, u, d, visited, path, path_Index)
19:    visited[u] ← True
20:    path.push_back(u)
21:    path_Index++
22:    if u == d then
23:        m.pus_back(path_Index)
24:        path.push_back(s)
25:        c.push_back(path);
26:    else
27:        for each i ∈ G.adjacency[s] do
28:            if !visited[i] and !marked[i] then
29:                Eccentricity(s,i,d,visited,path,path_index)
30:            end if
31:        end for
32:    end if
33:    path_Index−−
34:    visited[u] ← false
35: end function
36: function FACERETRIEVAL(Vectorofpathsasthedataelements)
37:    ConsideredPaths // Vector of paths
38:    IncludedCircuits // Vector of paths constructed
39:    IncludedVertex // Vector of Vertices already Included
40:    sort the c matrix with respect to all the path lengths.
41:    for each i ∈ c matix do
42:        for each vertex ∈ ith path considered do
43:            if !IncludedVertex[vertex] then
44:                Include the vertex and include the circuit
45:                Add the added Circuit to the ConsideredPath
46:                Add all the circuits considering the edges and the vertices of
   the Graph formed for counting the faces.
47:            end if
48:        end for
49:        if !IncludedPaths[i] then
50:            Include the circuit in the considered paths and the IncludedCir-
   cuits.
51:        end if
52:    end for
53: end function
```