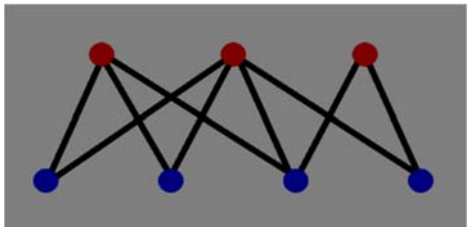
Project 4: Unweighted Graph

Write a program to check

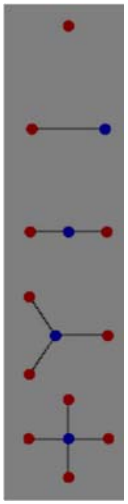
if a graph is bipartite

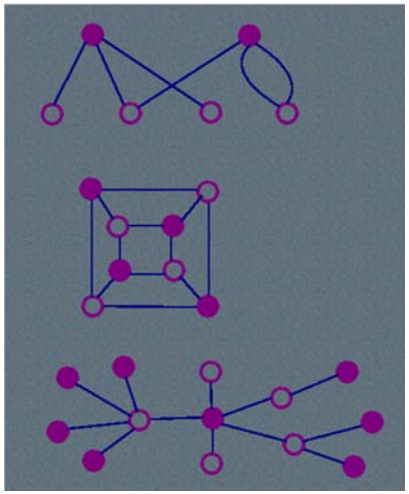
A bipartite graph

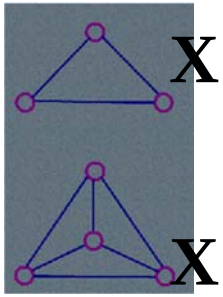
- Is unweighted graph
- G = (V,E)
- A set of vertices V can be partitioned into two subsets,V1 and V2, and no edge has both its vertices in the same subset
- The two sets **V1** and **V2** may be thought of as a coloring of the graph with two colors



Examples





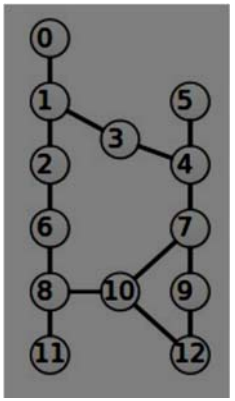


http://mathworld.wolfram.com/BipartiteGraph.html

http://www.personal.kent.edu/~rmuhamma/GraphTheory/MyGraphTheory/defEx.htm

Input

- a single command line argument giving **the name of a text file containing a graph**




```
0 1
1 0 2 3
2 1 6
3 1 4
4 3 5 7
5 4
6 2 8
7 4 9 10
8 6 10 11
9 7 12
10 7 8 12
11 8
12 9 10
```

Algorithm

1. Read a file containing a graph and store in a dynamic data structure

2. Traverse a graph and color vertices

- select a start vertex, and traverse a graph (BFS/DFS). If start at even vertex, i.e. 0, 2, etc., color it with **WHITE.** Otherwise, color it with **BLACK**

- while traverse a graph, color each vertex accordingly

- if we found color conflict between any two vertices during traversing, this is not a bipartite graph. Otherwise, it is a bipartite graph

Ways to do

```
class Vertex {
  bool isVisit;
  int color;
}
class Graph {
  vector <vector<Vertex>> vertices;
  void readAfile(filename) {}
  bool isBipartite() {}
}
```

```
class Graph {
  vector <vector<int>> vertices;
  int V; // number of vertices
  void readAfile(filename) {}
  bool isBipartite() {
    // store a color of each vertex,
    int colorArray[V];
    // store if a vertex has been visited
    bool isVisit[V];
  }
}
```

Note: we may use only one array (i.e. colorArray) and set 3 difference values, which represent VISITED, WHITE, BLACK.

isBipartite()

select a start vertex, u, color it

while (traverse a graph) **{**

  for each adjacent of u

    if it has not been colored

        color it with opposite color of u

    else

        check if there is a color conflict

}

Output

```
>prog g1.txt        >prog g2.txt
0 WHITE             0 WHITE
1 BLACK             3 BLACK
2 WHITE             2 WHITE
3 WHITE             4 WHITE
6 BLACK             1 BLACK
4 BLACK             6 BLACK
8 WHITE             5 BLACK
5 WHITE             7 BLACK
7 WHITE             10 BLACK
10 BLACK            8 WHITE
11 BLACK            9 WHITE
9 BLACK             11 WHITE
12 WHITE            CONFLICT 11 8
TRUE                FALSE
```