COLORING PROBLEM

BACKTRACKING

Coloring problem

- **NP-complete** problem !!! ~ exponential running time
- Problem: coloring the vertices of a graph such that no two adjacent vertices share the same color
- This is called a **vertex coloring**
- Reached popularity with the general public in the form of the popular number puzzle Sudoku
- The smallest number of colors needed to color a graph **G** is called its **chromatic number**
- There may be more than one solution: for example we can color a graph with **4** vertices in **12** ways with **3** colors !!!

APPLICATIONS

Bipartite graphs

- Determining if a graph can be colored with **2** colors is equivalent to determining whether or not the graph is bipartite, and thus computable in linear time using breadth-first search
- Bipartite graph: a graph whose vertices can be divided into two disjoint sets **U** and **V** (**U** and **V** are independent sets) such that every edge connects a vertex in **U** to one in **V**

Making schedules

We want to make an exam schedule for a university. We have different subjects and different students enrolled on every subject. Many subjects would have common students.

How do we schedule the exam so that no two exams with a common student are scheduled at the same time? How many minimum time slots are needed to schedule all exams?

This problem can be represented as a graph where every vertex is a subject and an edge between two vertices means there is a common student. So this is a graph coloring problem where minimum number of time slots is equal to the chromatic number of the graph

Radio frequency assignment

When frequencies are assigned to towers, frequencies assigned to all towers at the same location must be different because of the interference !!!

How to assign frequencies with this constraint? What is the minimum number of frequencies needed?

This problem is also an instance of graph coloring problem where every tower represents a vertex and an edge between two towers represents that they are in range of each other

Register allocation

In compiler optimization → register allocation is the process of assigning a large number of target program variables onto a small number of **CPU** registers

Map coloring

We want to construct a map of countries or states where adjacent countries or states can not be assigned the same color

This is the „tipical" coloring problem by the way !!!

Four colors are sufficient to color any map

„four color theorem"

Solutions

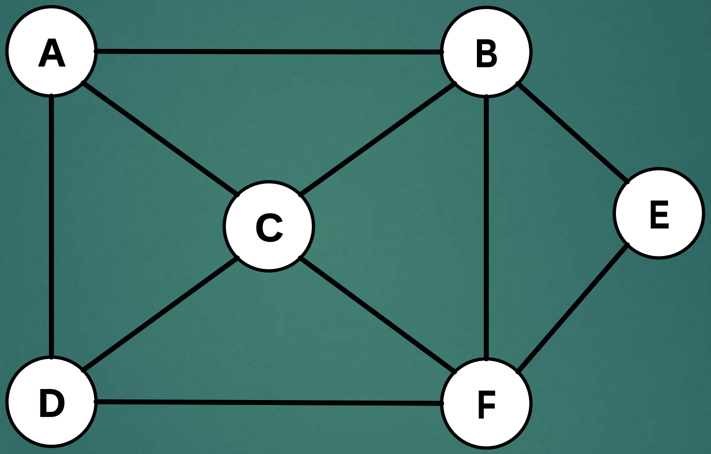
- Greedy approach → finds the solution but not the most optimal one

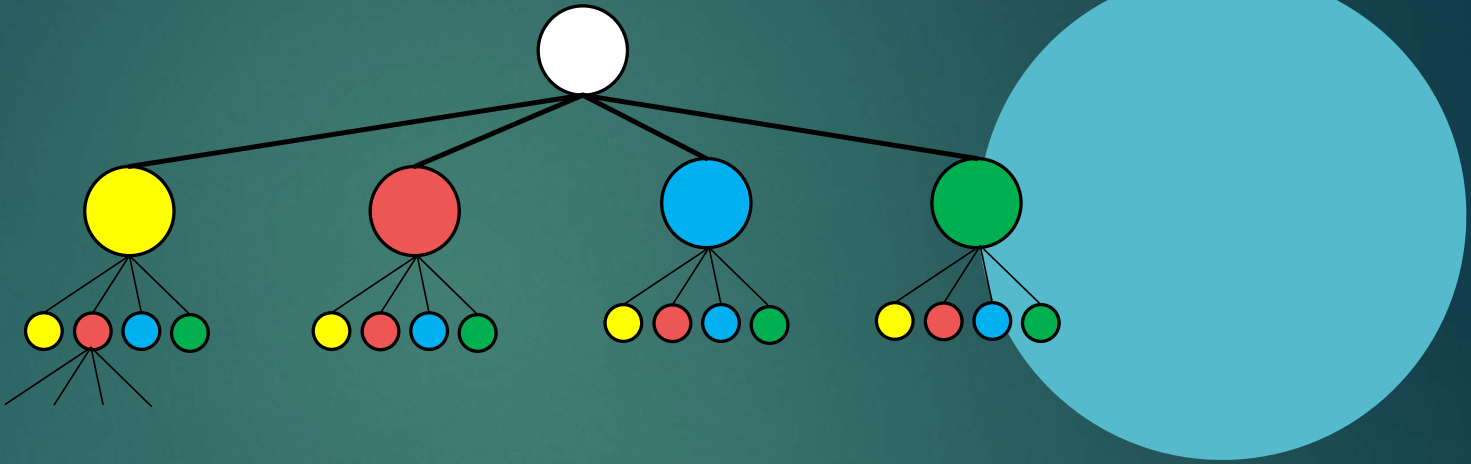
  It may uses more colors than necessary !!!

- Powell-Welsh algorithm → relies on sorting the nodes according to the degrees + we start assigning colors to the nodes with the most neighbors !!!
- **BACKTRACKING**

Backtracking

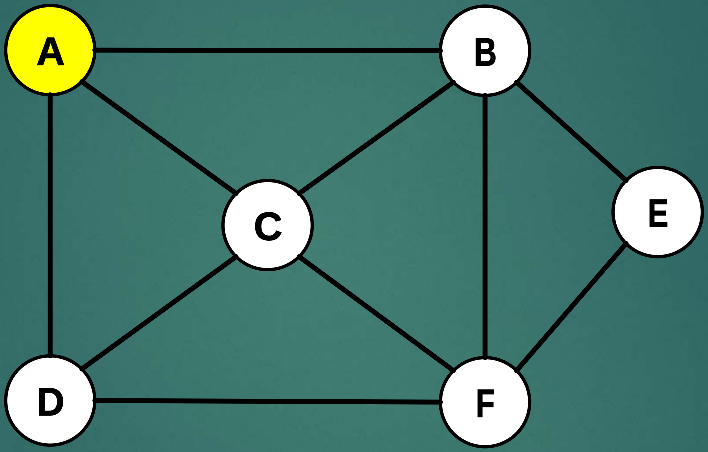
- We assign colors one by one to different vertices starting from the first vertex (optional)
- Before assigning a color → we check for safety by considering already assigned colors to the adjacent vertices
- If we find a color assignment which is feasible → we mark the color assignment as part of solution
- If we do not a find color due to clashes → we backtrack !!!

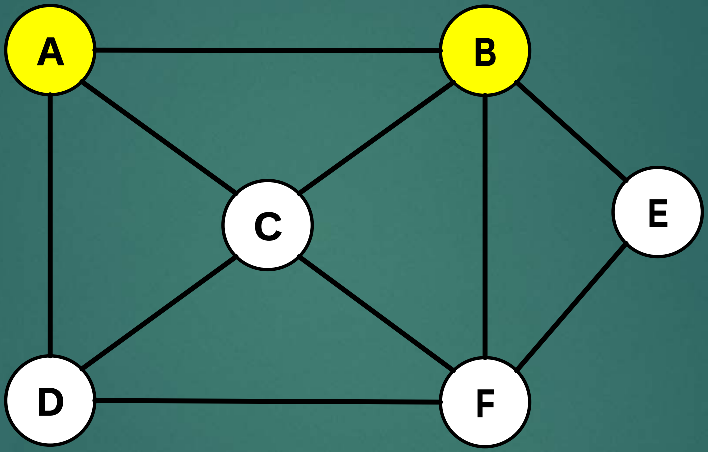

A
B
C
D
E
F

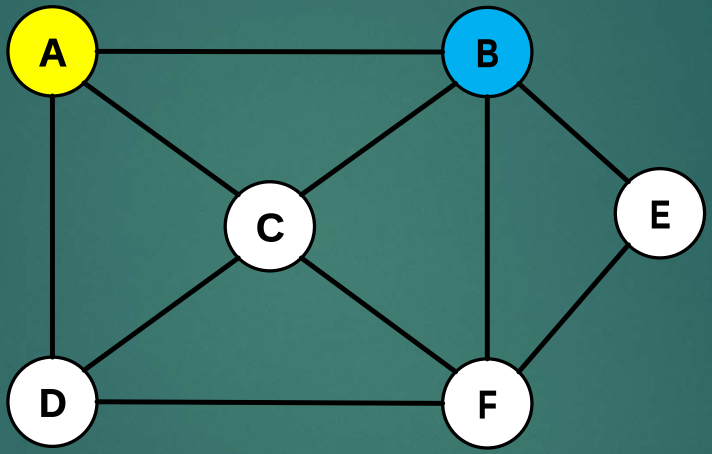


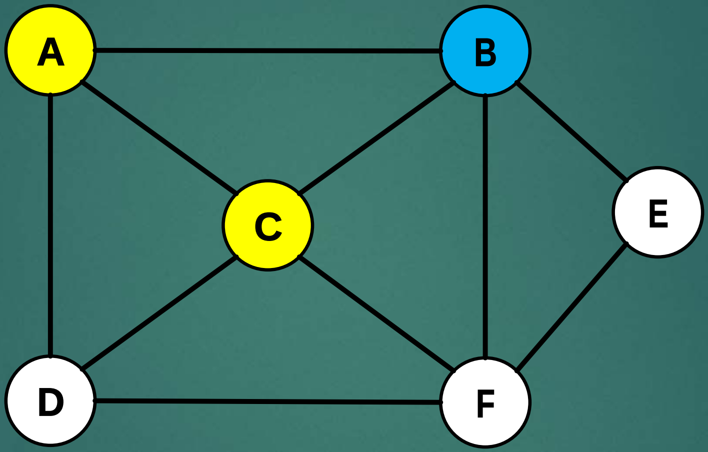
vertex 0

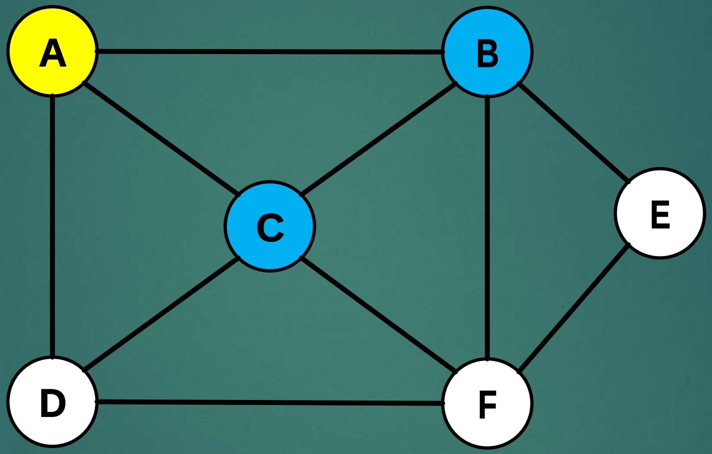
vertex 1

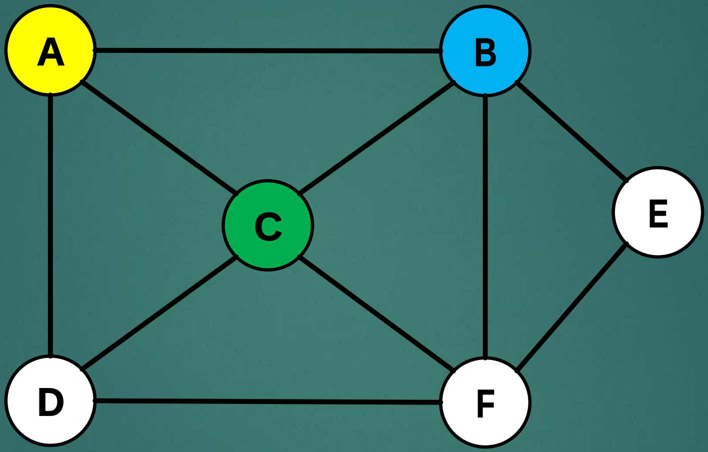

A
B
C
E
D
F

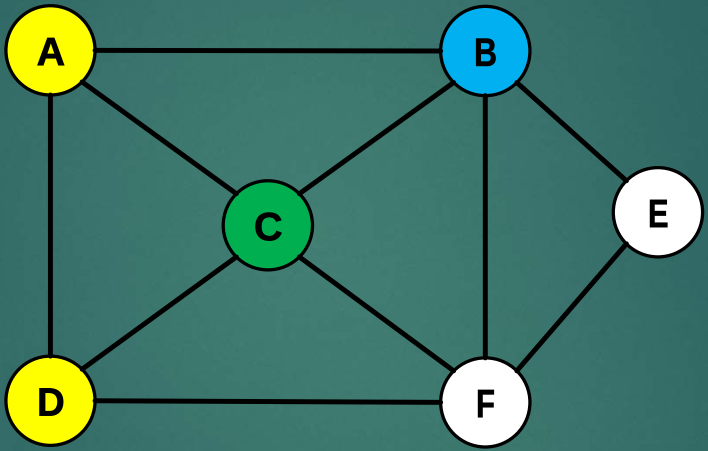

A
B
C
E
D
F

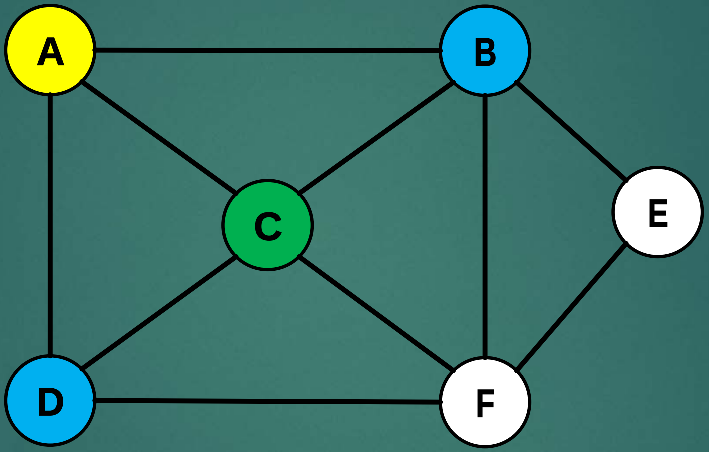

A
B
C
E
D
F

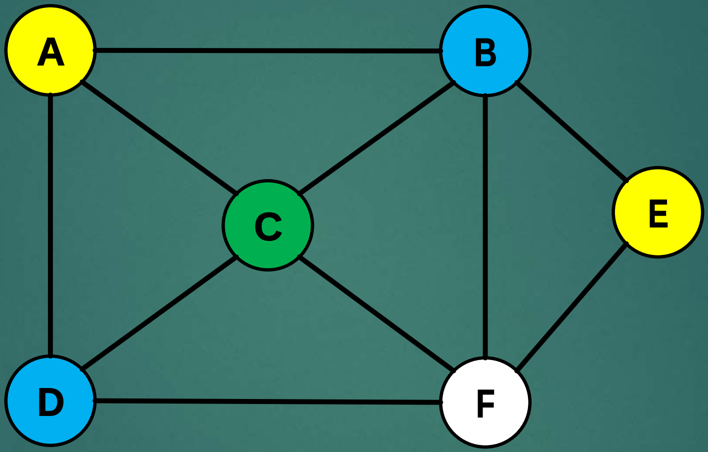

A
B
C
E
D
F

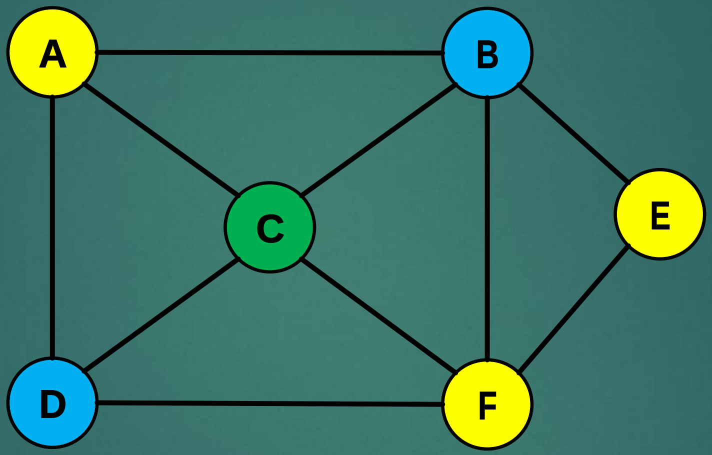

A
B
C
D
E
F

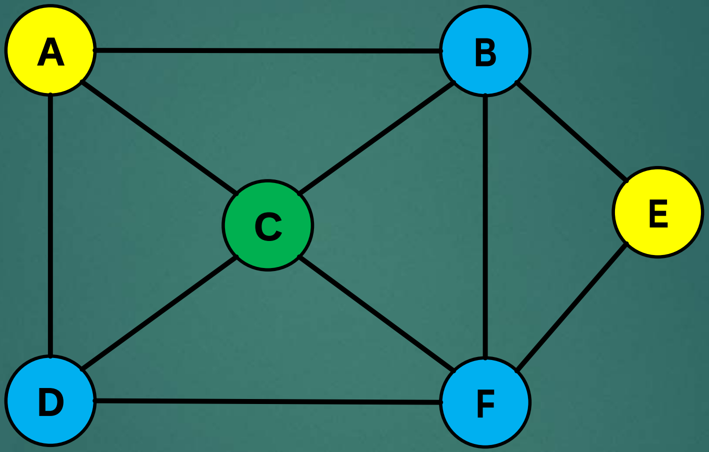

A
B
C
D
E
F

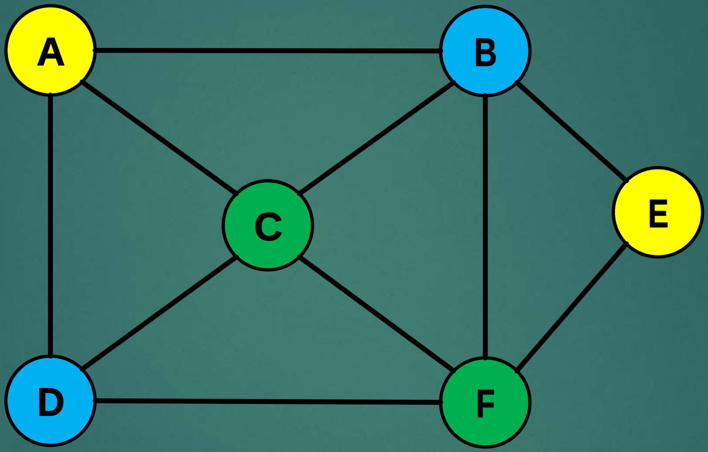

A
B
C
E
D
F

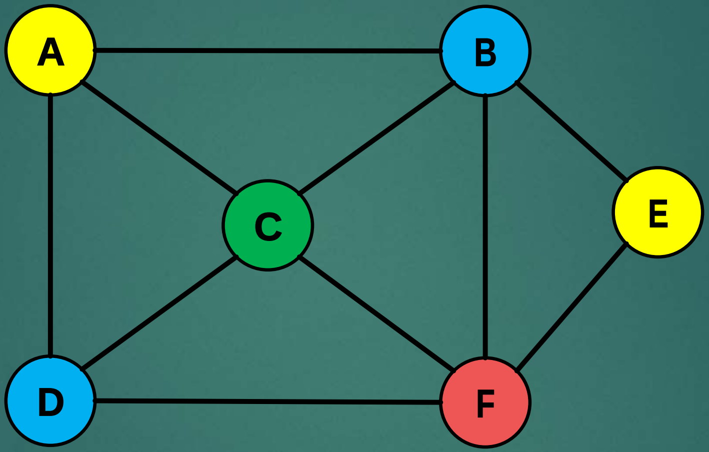

A
B
C
D
E
F

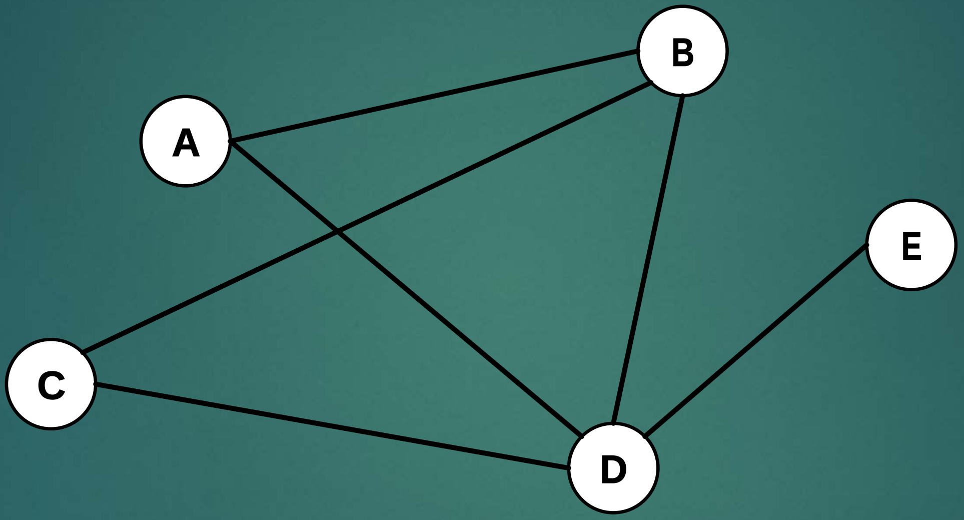

A
B
C
D
E
F


A
B
C
E
D
F

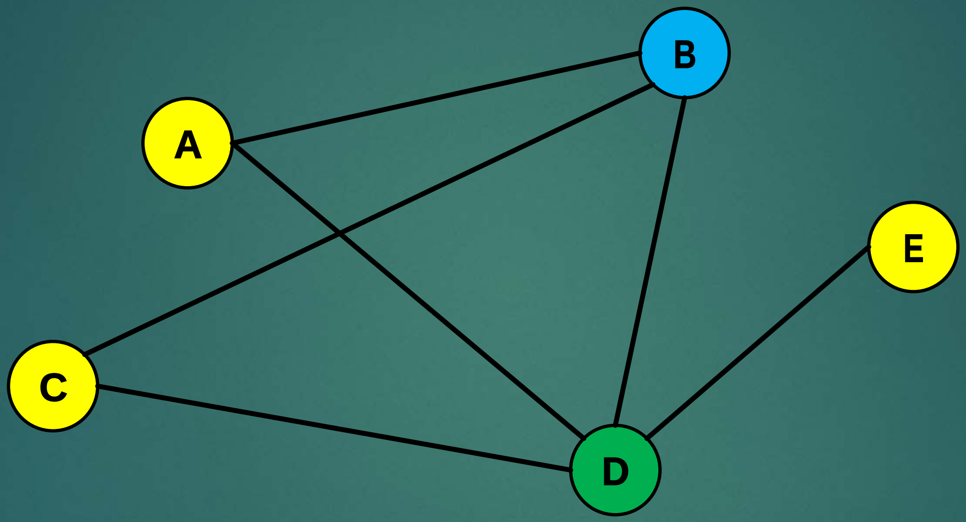

A
B
C
D
E
F


A
B
C
D
E
F


A
B
C
E
D
F



A
B
C
D
E


A
B
C
D
E