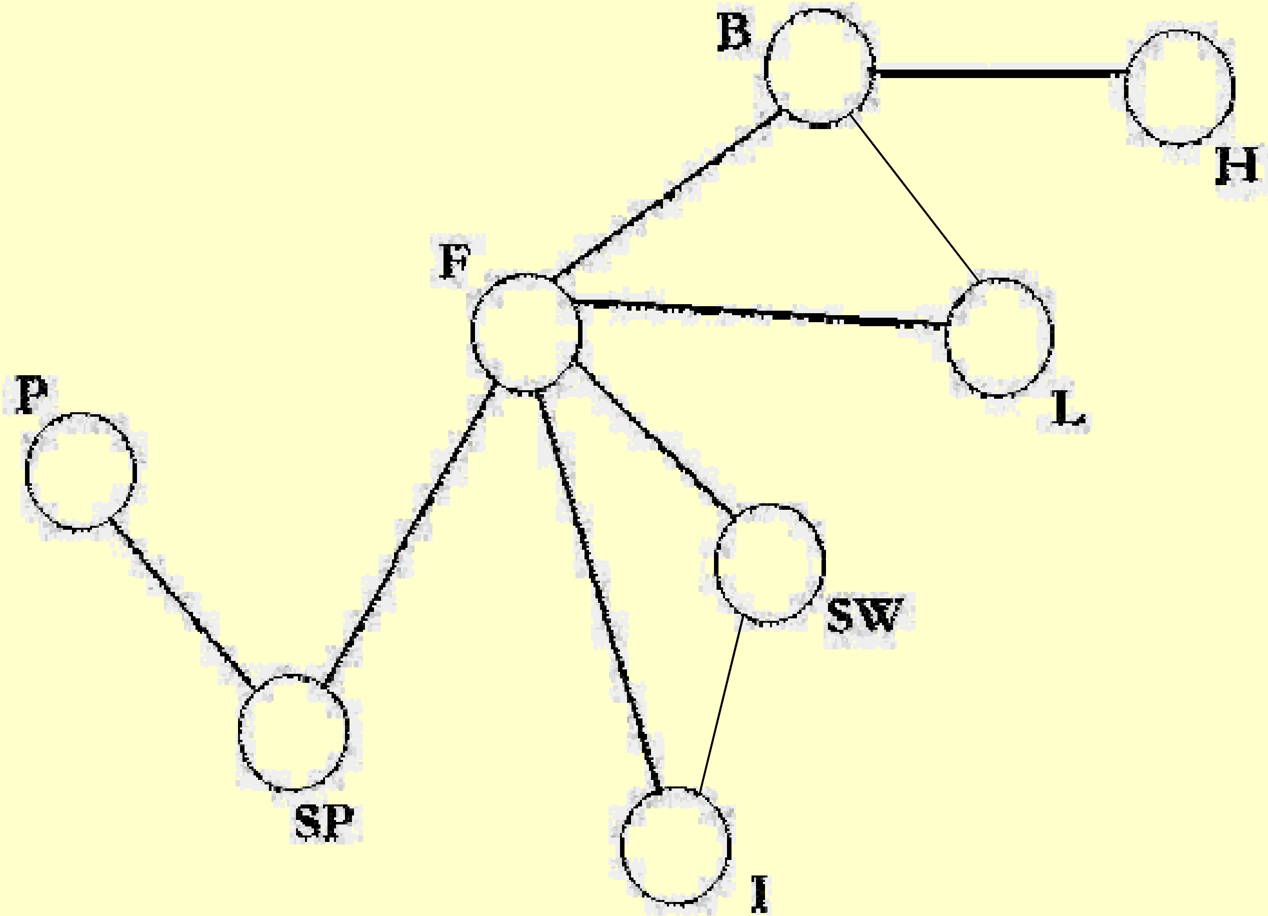
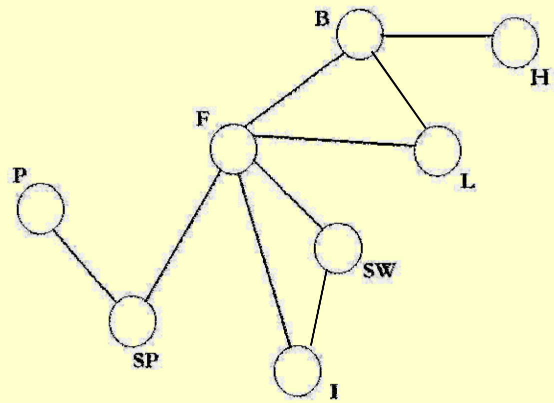
Número Cromático

Problema

- Quantas cores são necessárias para pintar este mapa?

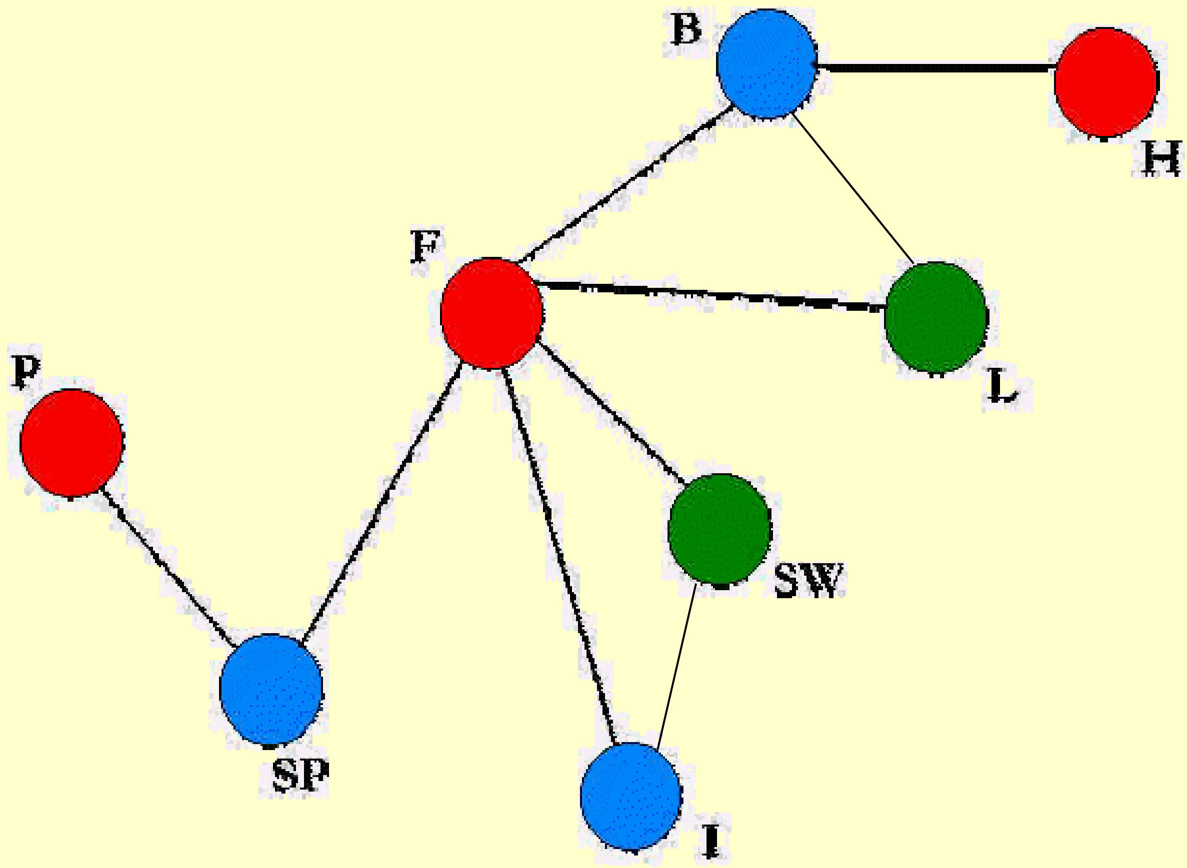
Problema

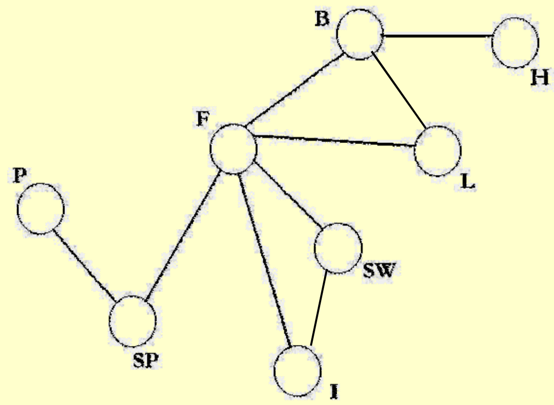

B
H
F
L
P
SW
SP
I


B
H
F
L
P
SW
SP
I


Holland
Belgium
Luxembourg
Switzerland
Portugal
France
Spain
Italy

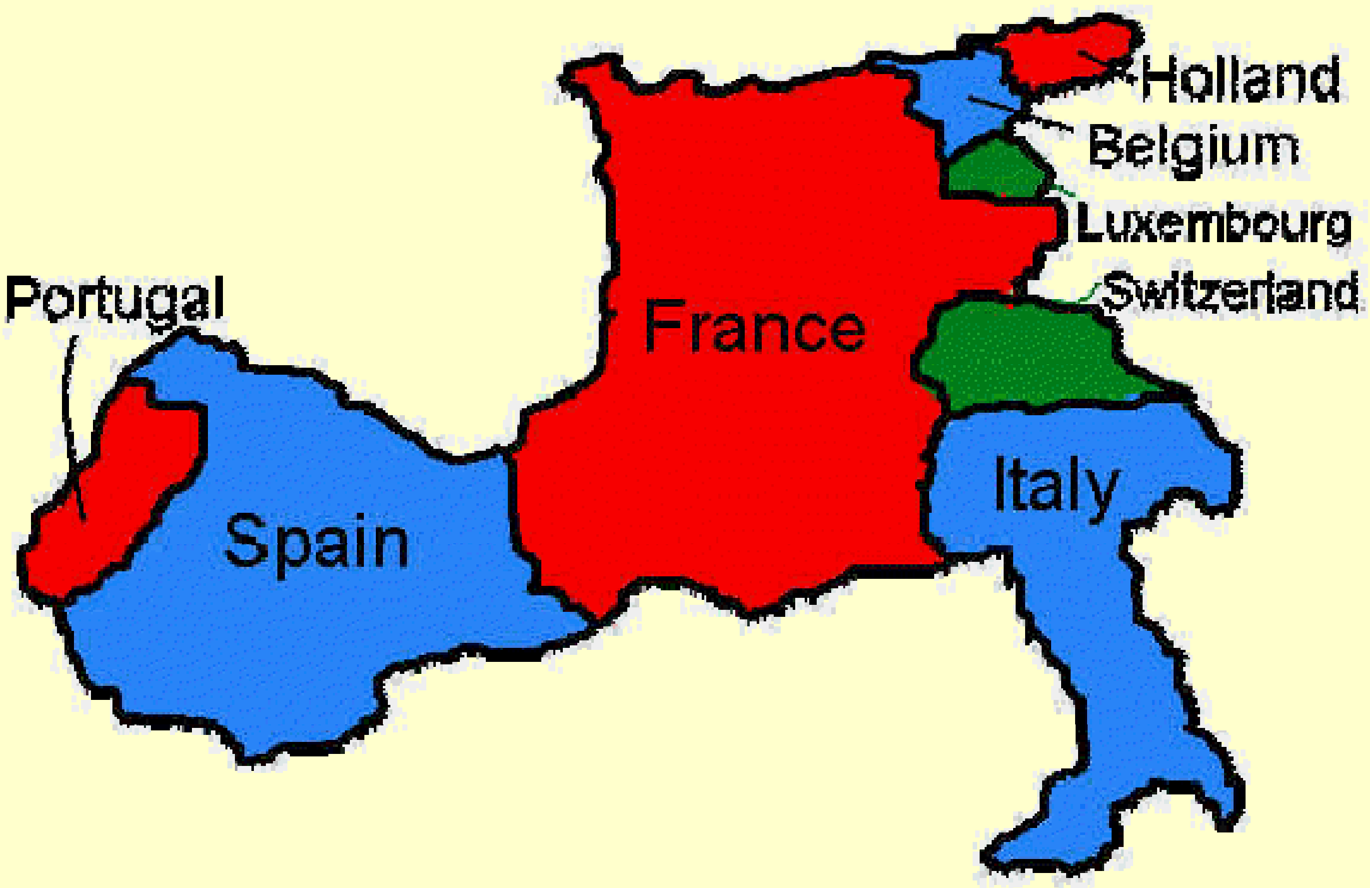
Problema


B
H
F
L
P
SW
SP
I


B
H
F
L
P
SW
SP
I


Holland
Belgium
Luxembourg
Switzerland
Portugal
France
Spain
Italy

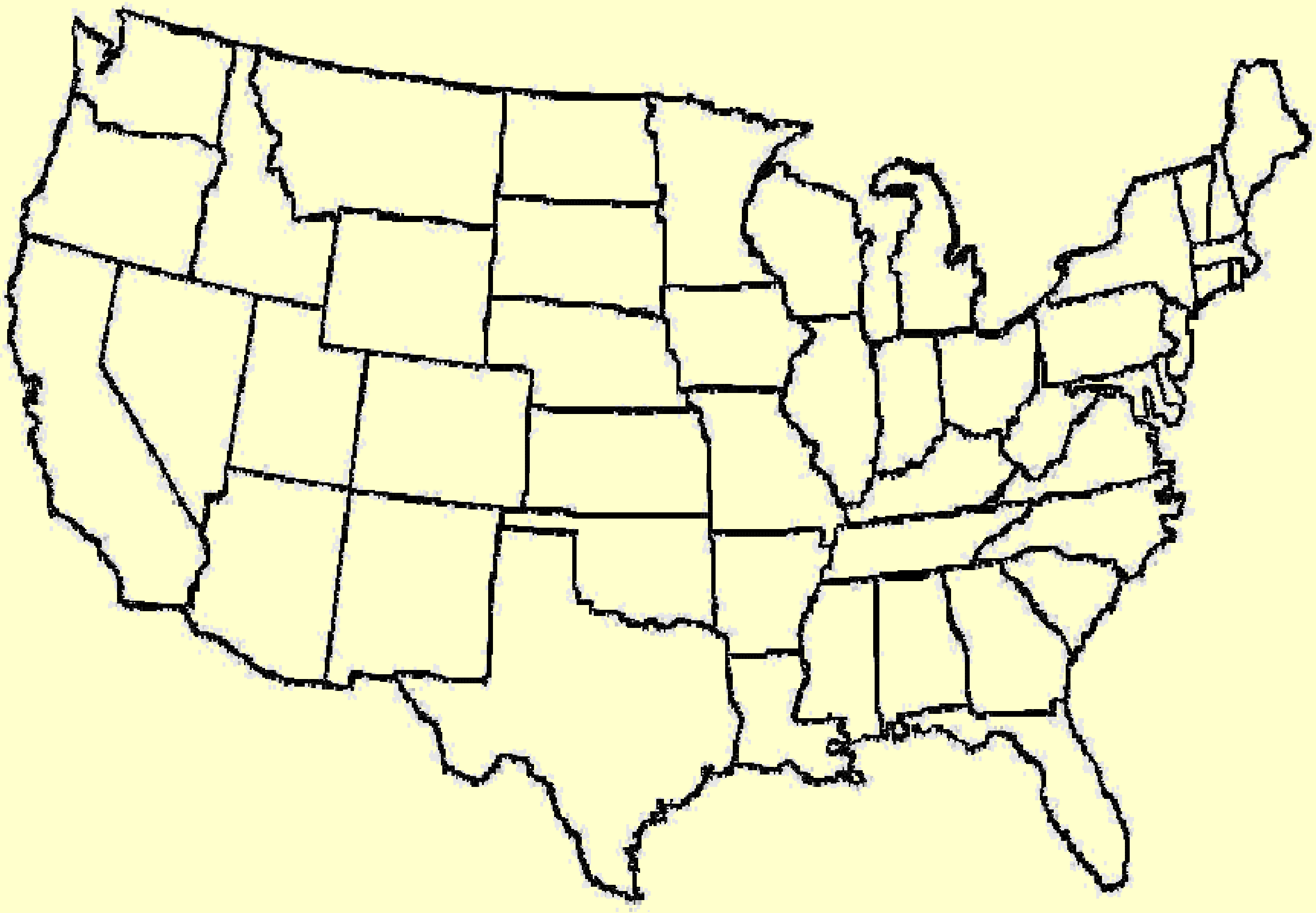
Problema



Holland
Belgium
Luxembourg
Switzerland
Portugal
France
Italy
Spain

Um problema mais complicado

- E este?



Problema:

- Uma companhia manufatura os produtos químicos C1, C2, … Cn. Alguns destes produtos podem explodir se colocados em contato com outros. Como precaução contra acidentes,a companhia quer construir k armazéns para armazenar os produtos químicos de tal forma que produtos incompatíveis fiquem em armazéns diferentes.
  - Qual é o menor número k de armazéns que devem ser construídos? Como resolver este problema com a ajuda da teoria dos Grafos?

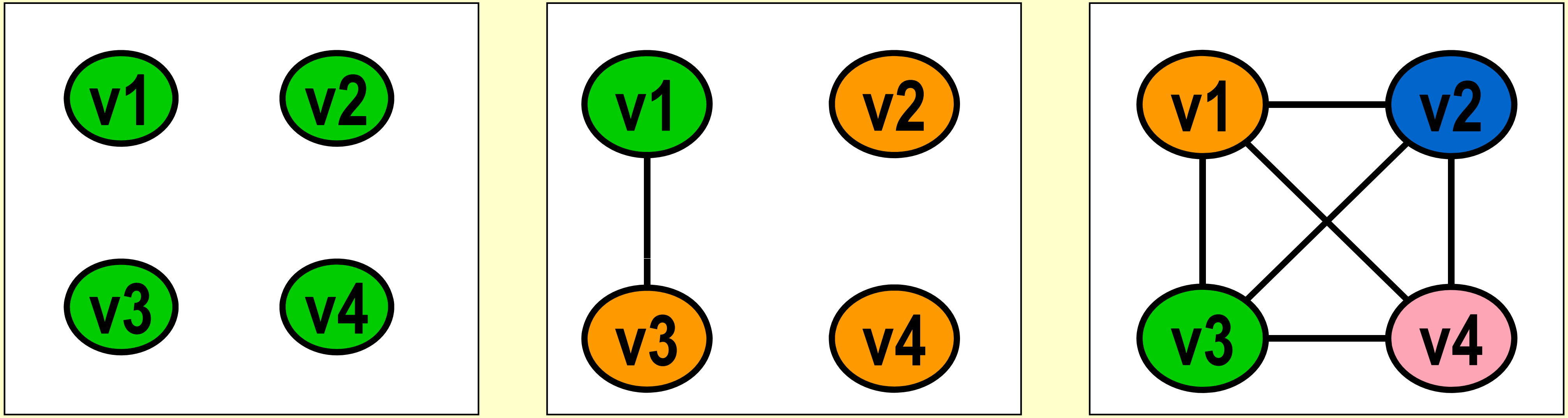
Problema:

- Emissoras de televisão vão ser instaladas em estações baseadas em nove cidades de nosso estado (cidades A, B, ..., I). As regulamentações do setor de telecomunicações indicam que uma mesma emissora não pode ser instalada em duas cidades com distância inferior a 150Km.
  – Considere a tabela abaixo que indica as distâncias entre as cidades. Qual o menor número de emissoras para contemplar as nove cidades? Utilize a teoria dos grafos para resolver este problema e justificar a sua resposta.

	A	B	C	D	E	F	G	H
B	85							
C	137	165						
D	123	39	205					
E	164	132	117	171				
F	105	75	235	92	201			
G	134	191	252	223	298	177		
H	114	77	113	117	54	147	247	
I	132	174	22	213	138	237	245	120

Número cromático
Coloridos ótimos

- menor número de cores necessárias para colorir todos os vértices de um grafo, de modo que nodos adjacentes tenham cores diferentes



- O **Teorema das 4 Cores**, provado em 1976, se constitui em um dos resultados mais importantes da matemática no Século XX. (permaneceu sem solução desde 1852)
- **Aplicação em muitos problemas práticos**
- Uma coloração de um grafo é uma atribuição de cores aos vértices, de modo que vértices adjacentes tenham cores distintas.
  - Um grafo *G tem k-coloração se ele pode ser colorido* com *k cores.*

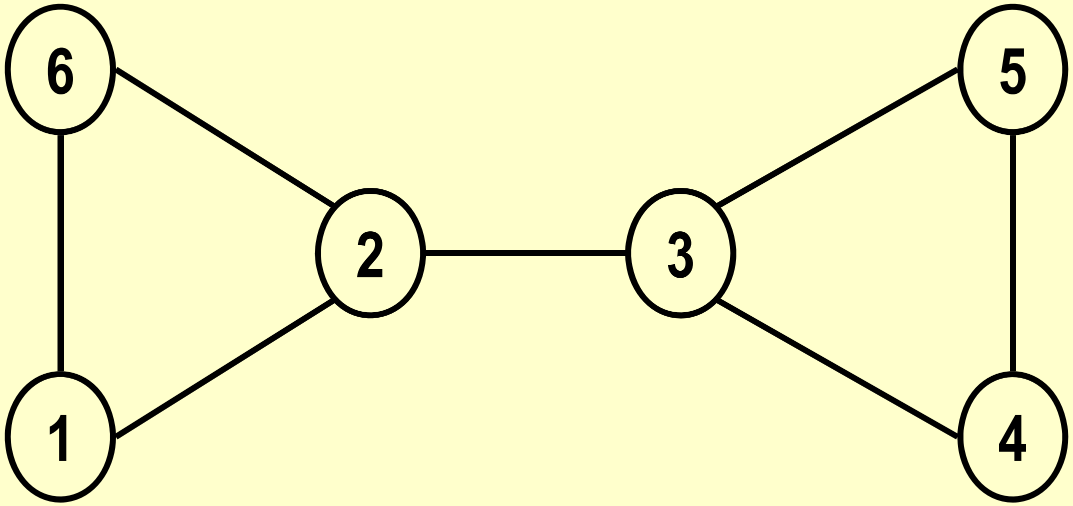
- Se *G tem k-coloração mas não pode ter (k-1)-* coloração:
  - O número cromático de *G é k.*
  - *G é k-cromático.*
  - $\chi(G) = k$

Cliques

Quando todos os pares de vértices são adjacentes

Um clique é um subgrafo completo máximo

Não contido em outro subgrafo completo



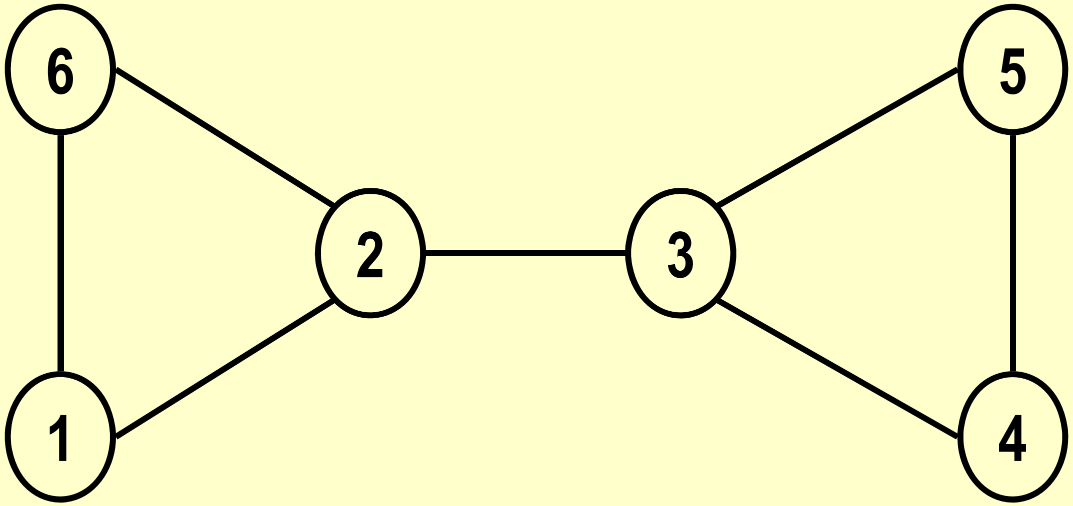
Cliques

Quando todos os pares de vértices são adjacentes

Um clique é um subgrafo completo máximo

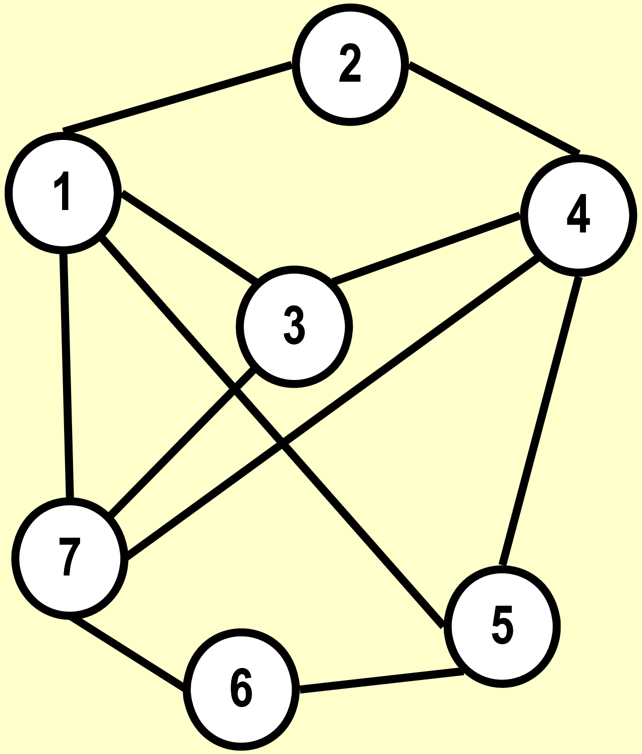
Não contido em outro subgrafo completo

~~{1 6}~~



{1 2 6}
{2 3}
{3 4 5}

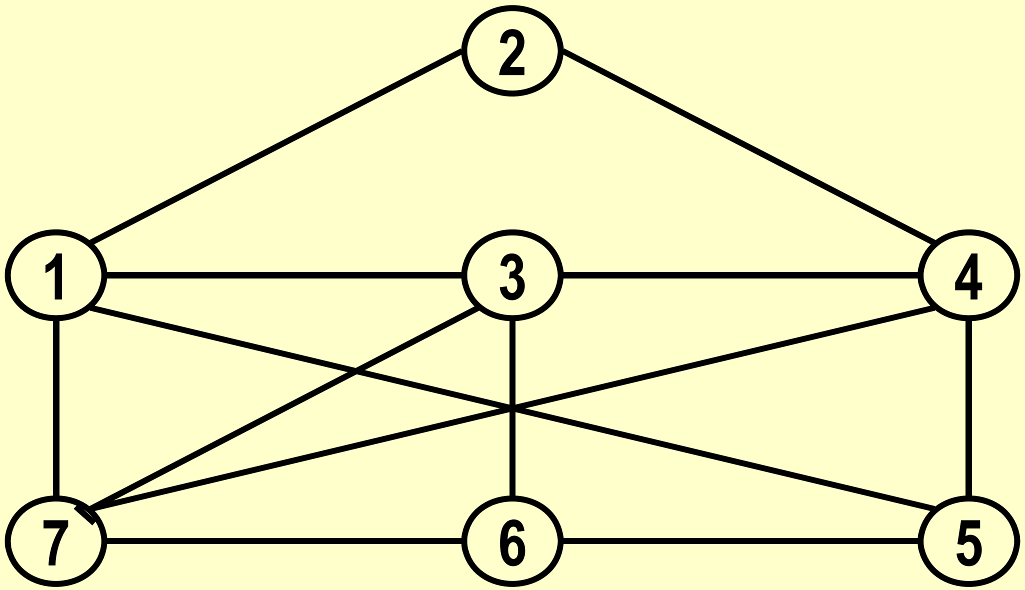
Cliques



Conjunto de vértices independentes máximo (CVIM)

Vértices não adjacentes entre si

Não contido em outro conjunto de vértices interiormente estável



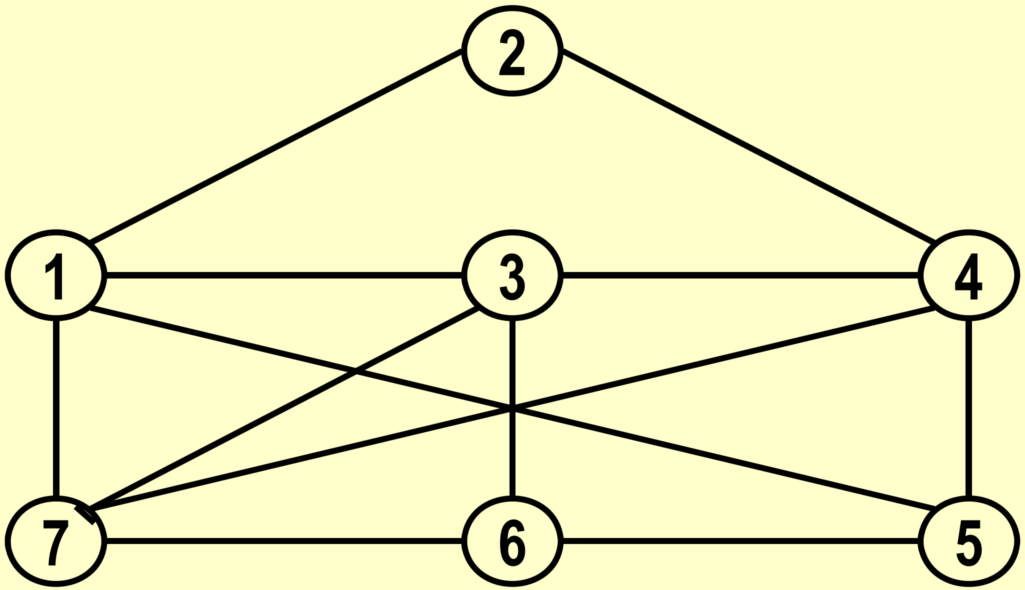
Conjunto Independente (maximal)

Conjunto de vértices independentes máximo (CVIM)

Vértices não adjacentes entre si

Não contido em outro conjunto de vértices interiormente estável

~~{2 3}~~



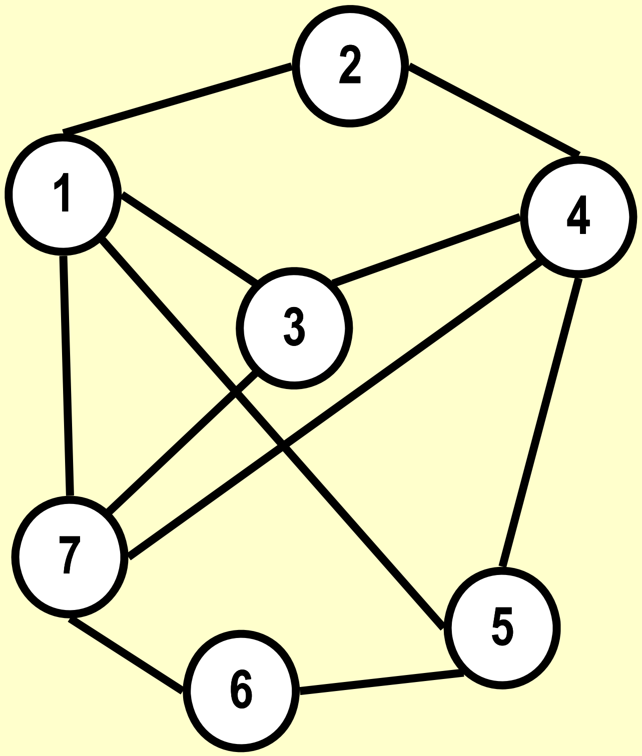
S1 = {1 4 6}

S2 = {2 3 5}

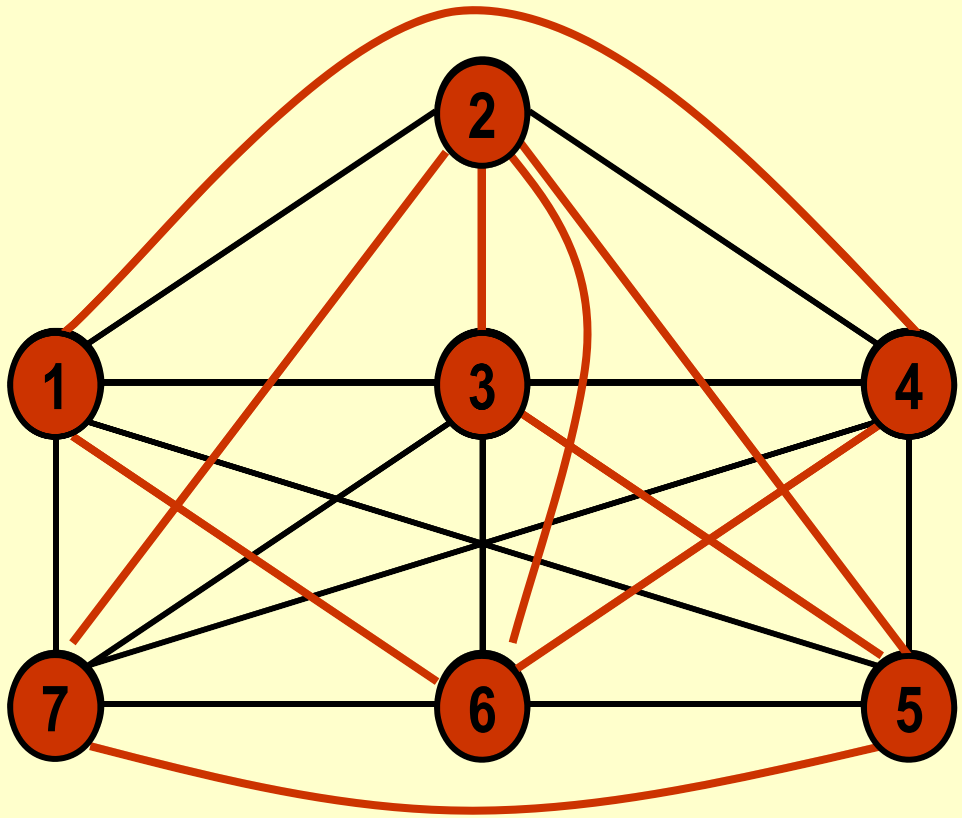
S3 = {2 5 7}

S4 = {2 6}

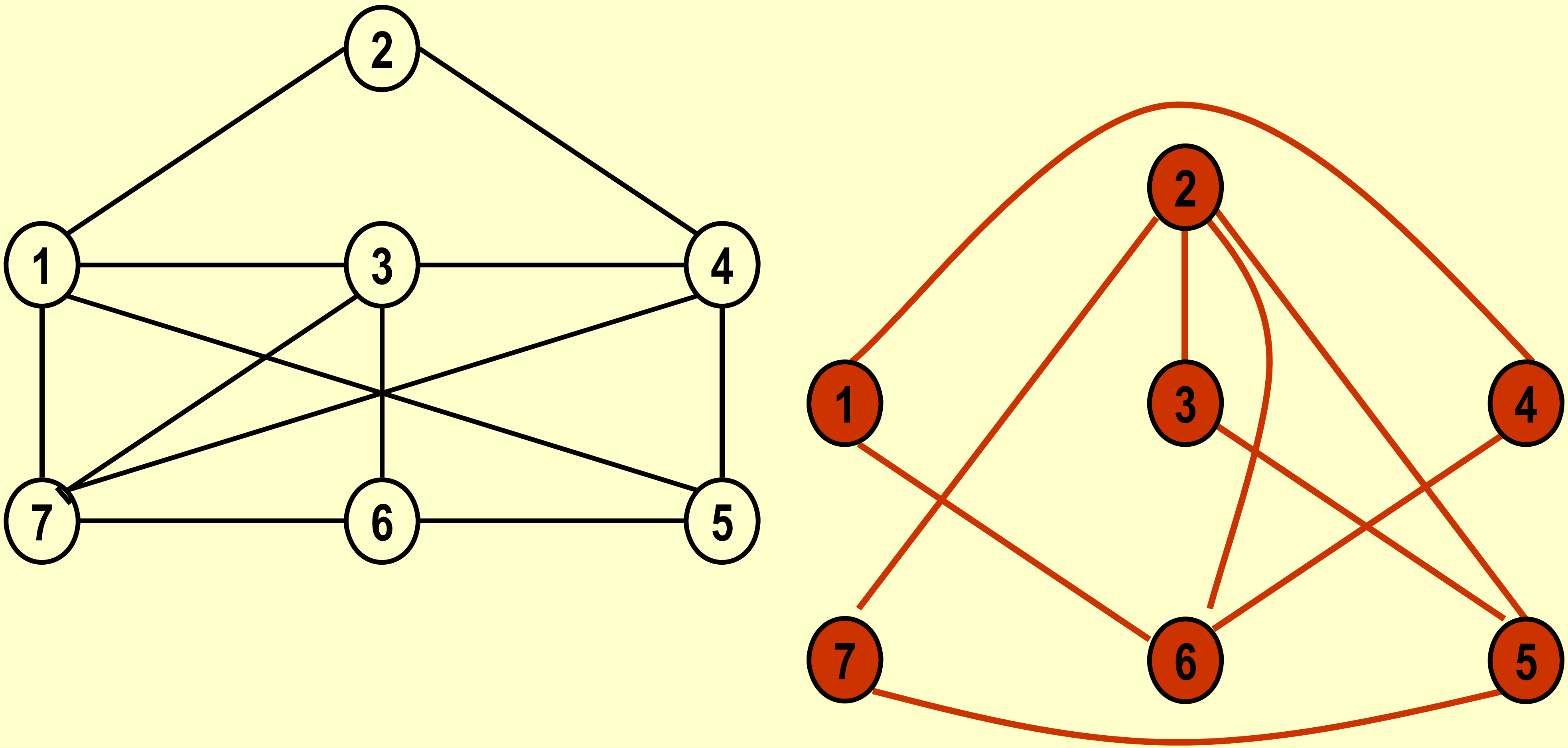
Conjunto de vértices independentes máximo (CVIM)



Grafo complemento de um grafo

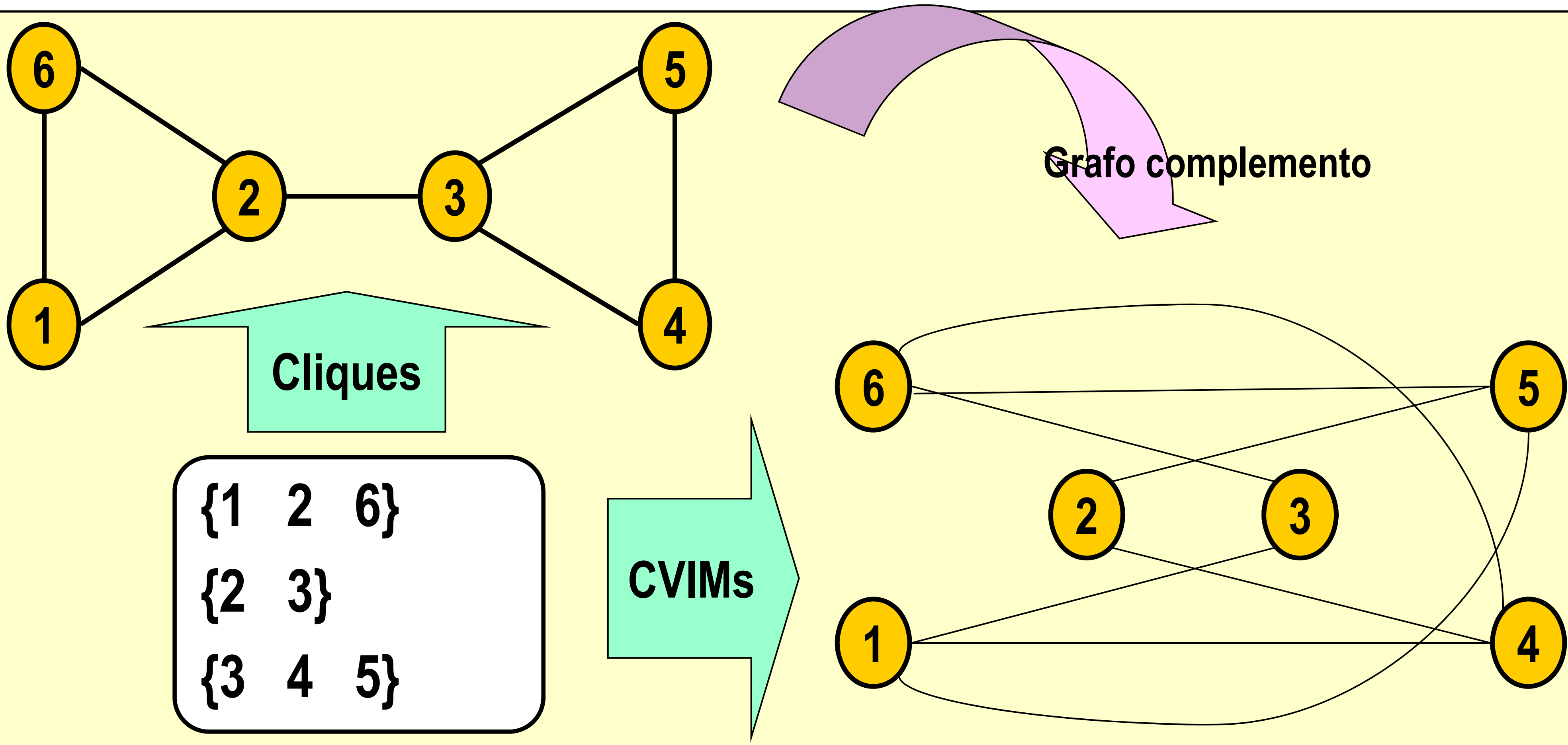


Grafo complemento de um grafo



Cliques x CVIMs

Os CVIMs são os cliques do grafo complemento, para grafos não orientados

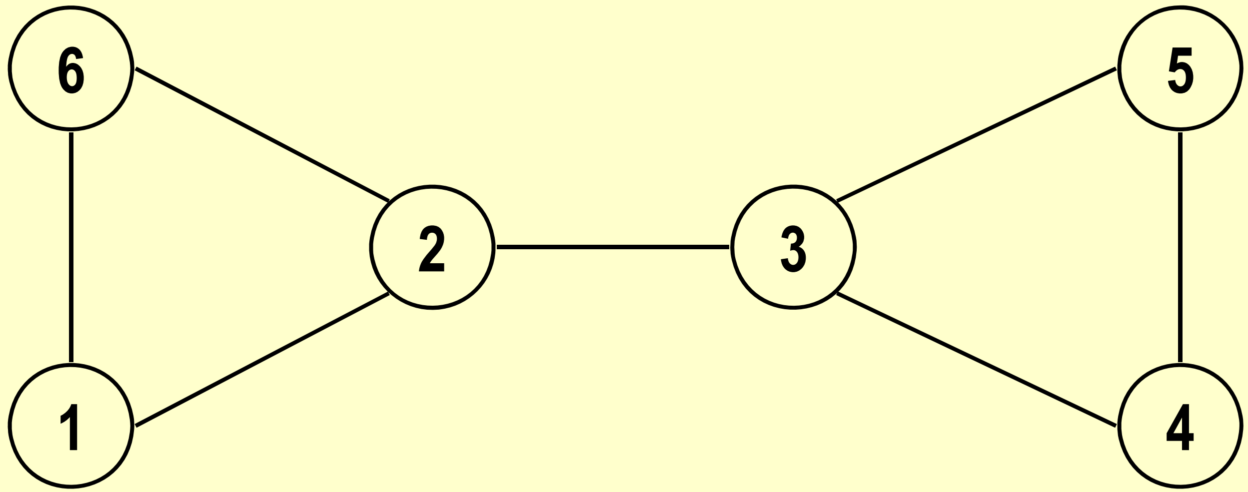


Algoritmo para gerar os cliques de um grafo

```
Procedimento cliques()
// VC: Conjunto de vértices de um clique
// CAND: Candidatos a inclusão em VC
// ANT: Vértices já esgotados
início
 VC ← {};
 ANT ← {};
 CAND ← V(G);
 rCliques(VC, CAND, ANT)
fim; // cliques
```

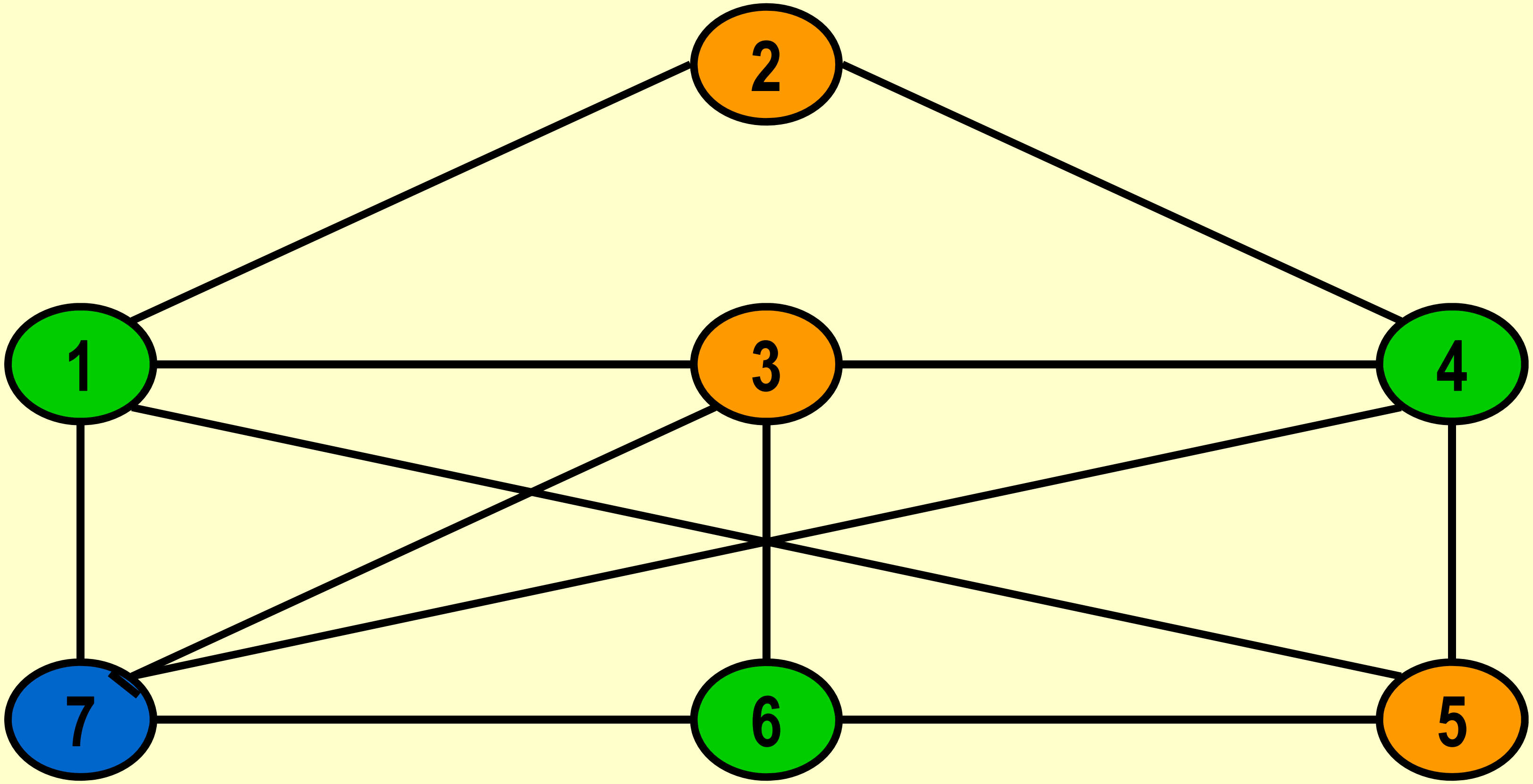
```
Procedimento rCliques (Conjunto VC, Conjunto CAND,
Conjunto ANT)
// Determina os cliques de G
// Se todos os candidatos foram examinados e VC é
// independente dos cliques já formadas,então VC é um clique
Vértice v;
Conjunto VC1, CAND1, ANT1;
início
   se CAND = {} e ANT = {}
   então *** Encontrou  Clique  VC ***;
   // Examina candidato para estender VC
   senão enquanto CAND  ≠  {}
            faça início
               v ←CAND.escolhe();
               VC1 ←VC ∪ {v};
               CAND1←CAND ∩ adj(v);
               ANT1←ANT ∩ adj(v);
               rCliques(VC1,CAND1,ANT1);
               CAND←CAND - {v};
               ANT←ANT ∪ {v};
            fim
fim; // rCliques
```

Exemplo




```
<VC, ANT, CAND>
   v=2  <{2}, {}, {1,3,6}>
      v=1  <{2,1}, {}, {6}>
         v=6  <{2,1,6},{},{}>    Clique {2,1,6}
           <{2,1},{6},{}>
        <{2},{1},{3,6}>
           v=3  <{2,3},{},{}>    Clique {2,3}
           v=6  <{2,6},{1},{}>
```

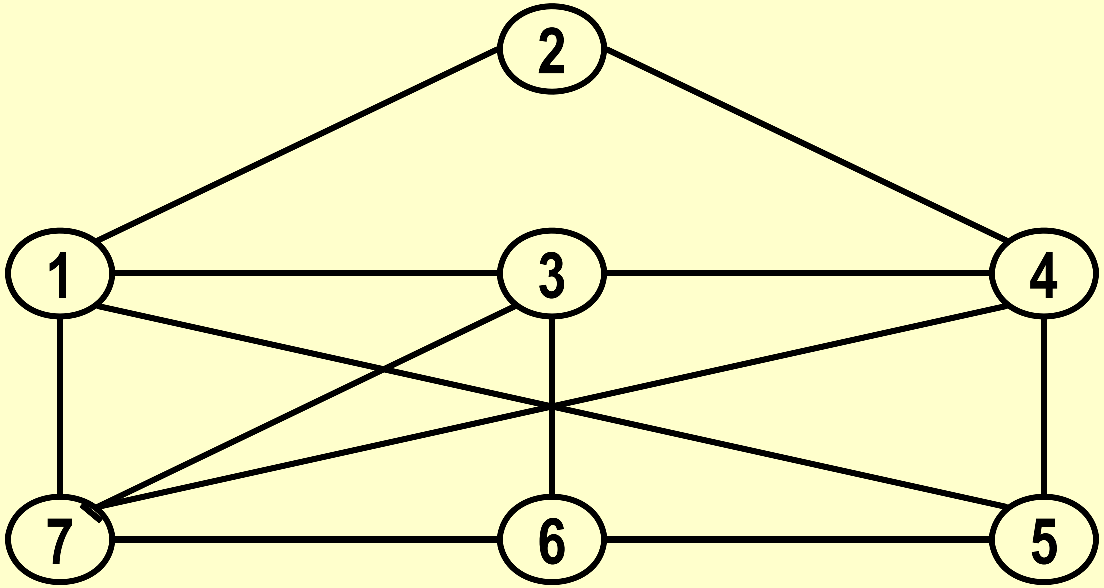
Número cromático??



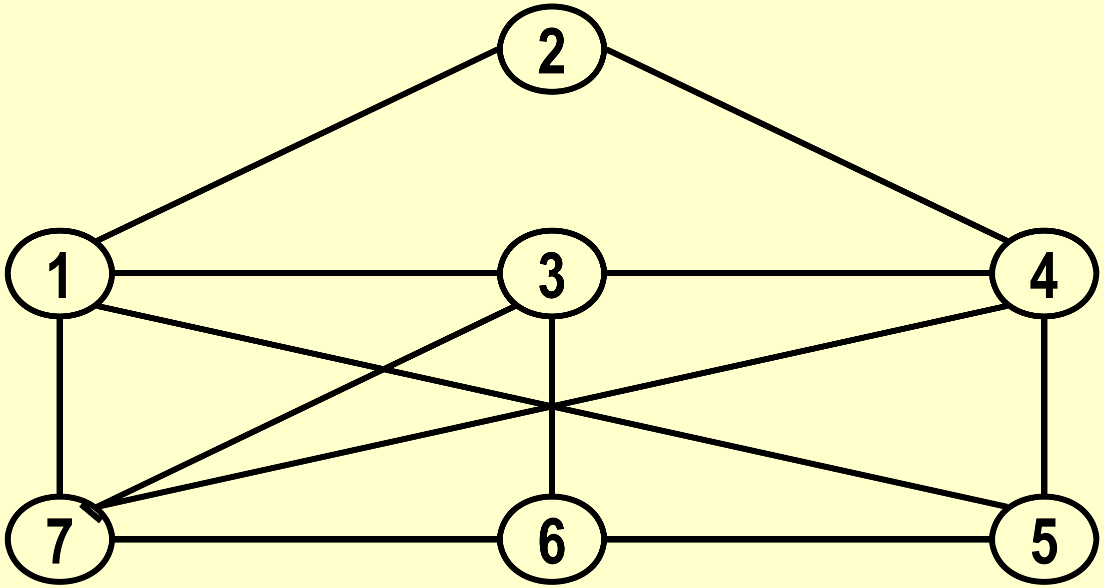
Algoritmo para determinar o número cromático de um grafo

1. Determinar os conjuntos de vértices independentes máximos (CIVMs) (cliques do grafo complemento)
2. Escolher um dos conjuntos – analisar os vértices que sobram
   Se existe algum conjunto dentro deles, pode ser de outra cor
   Repetir até esgotar todos os vértices
2. S_{ij} – conjunto a ser colorido com uma cor

Exemplo de determinação do número cromático de um grafo



Exemplo de determinação do número cromático de um grafo



CVIMs:

S_{11}={1,4,6}

S_{12}={2,3,5}

S_{13}={2,7,5}

S_{14}={2,6}

S_{11}={1,4,6} sobra: $\{\ \underbrace{2,\ 3,\ 5,}_{S_{12}}\ 7\ \}$

S_{21}={1,4,6 | 2,3,5} sobra: { 7 }

S_{31}={1,4,6 | 2,3,5 | 7}

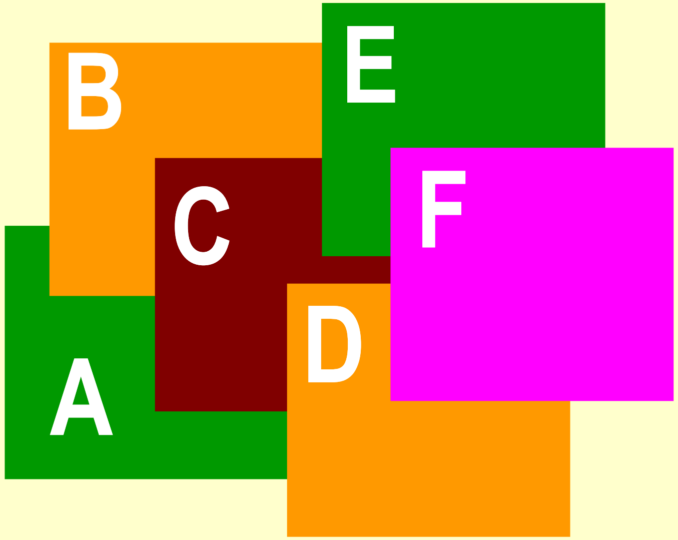
1 possibilidade

3 cores

Exemplo de determinação do número cromático de um grafo

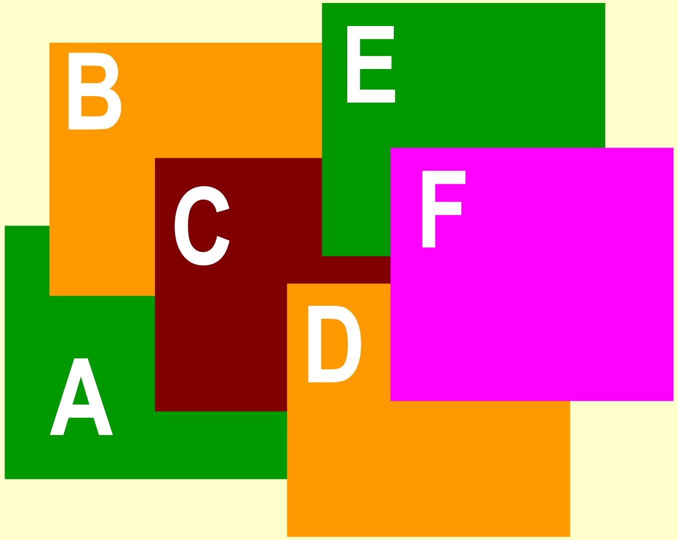
{1,4,6}	{1,4,6|2,3,5}	{1,4,6|2,3,5|7}	# 3
	{1,4,6|2,7,5}	{1,4,6|2,7,5|3}	# 3
{2,3,5}	{2,3,5|1,4,6}	{2,3,5|1,4,6|7}	# 3
{2,7,5}	{2,7,5|1,4,6}	{2,7,5|1,4,6|3}	# 3
{2,6}	{2,6|1|3|4|5|7}		# 6

Algoritmos



Procurar um número *próximo* ao número cromático!!!!

Algoritmos

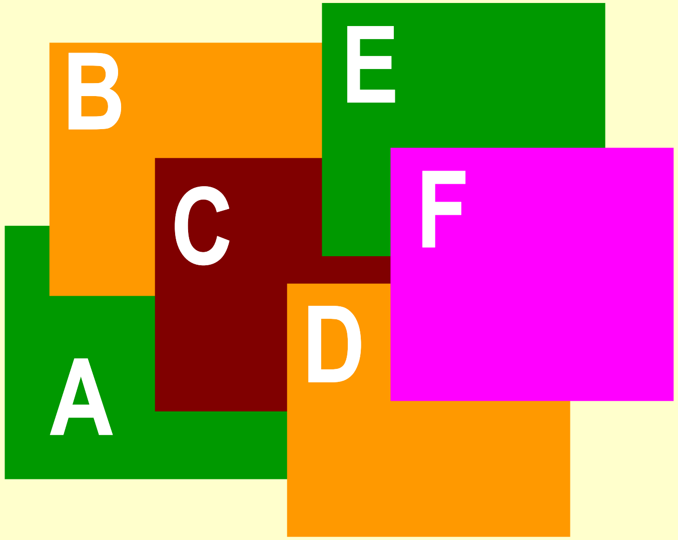


Solução : <u>lista de adjacências</u>

– vetor com as regiões que devem ser coloridas;

A
B
C
D
E
F

Algoritmos

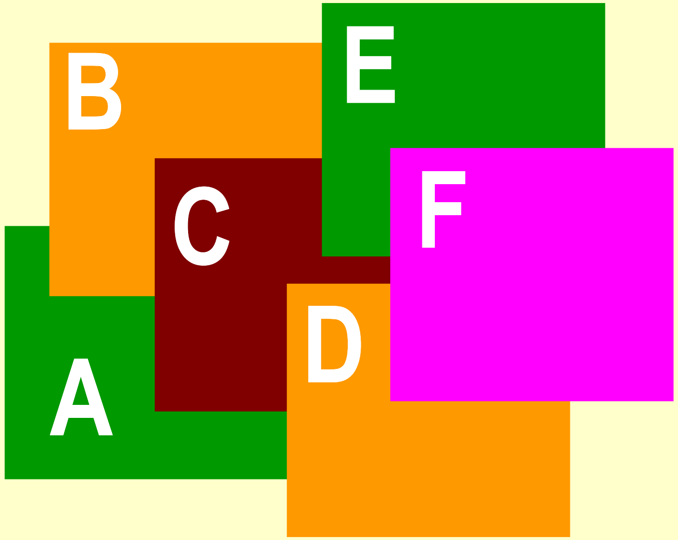


Solução : <u>lista de adjacências</u>

- **vetor** com as regiões que devem ser coloridas;
- **lista** com os demais elementos que são as *regiões adjacentes* a este.

A [B, C, D]
B [A, C, E]
C [A, B, D, E, F]
D [A, C, F]
E [B, C, F]
F [D, E C]

Algoritmos



– **Escolhe-se uma região inicial (exemplo: A) e atribui uma cor;**

– **Para atribuir cor a B, é verificado se dentre as cores existentes, existe uma que não esteja colorindo nenhuma região adjacente a B, então essa cor deverá ser escolhida. Se todas as cores existentes estiverem sendo utilizadas em regiões vizinhas a B, então uma nova cor é criada.**

o raciocínio é repetido analogamente para cada uma das regiões subsequentes.

Busca em Profundidade

Programa Principal

Montar a lista de adjacências

Inicializar a estrutura de cores

Escolher o vertice *Vi* de maior grau para ser colorido primeiro

Chamar a sub-rotina Colore_Vertice para colorir o vertice *Vi* escolhido

Sub-rotina COLORE-VERTICE

se o vertice *Vk* ainda nao foi colorido

    procurar a cor *C* apropriada

    se nao existir cor apropriada para colorir o vertice *Vk*

        criar uma nova cor *C*

    fim se

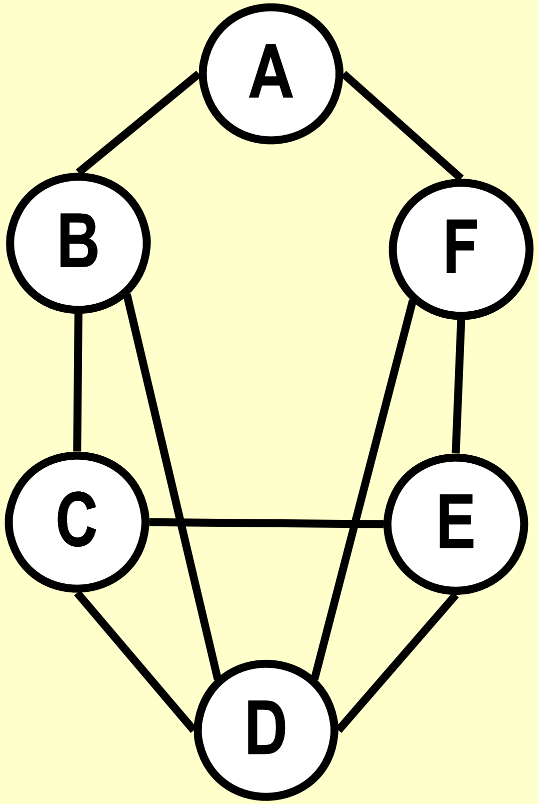
    colorir o vertice *Vk* com a cor *C*

    para todo vertice *Vj* adjacente a *Vk* faça

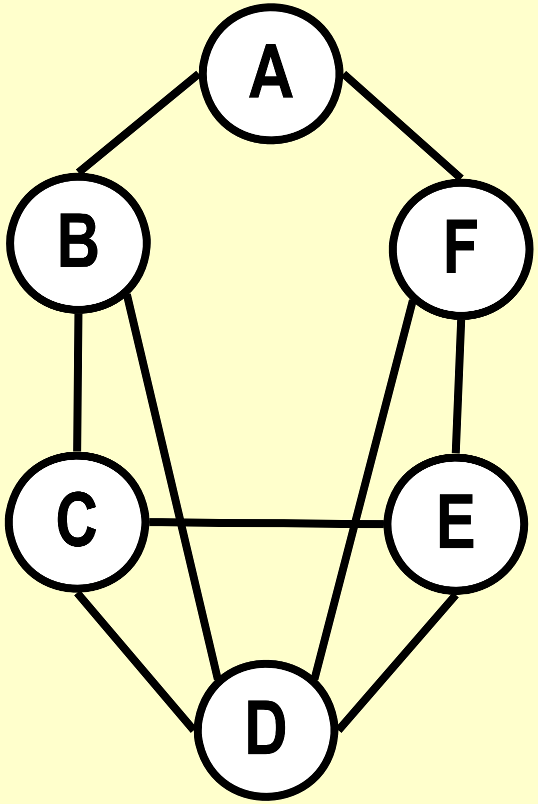
        chamar a sub-rotina Colore_Vertice para colorir o vertice *Vj*

fim se

Busca em Profundidade



Busca em Profundidade



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

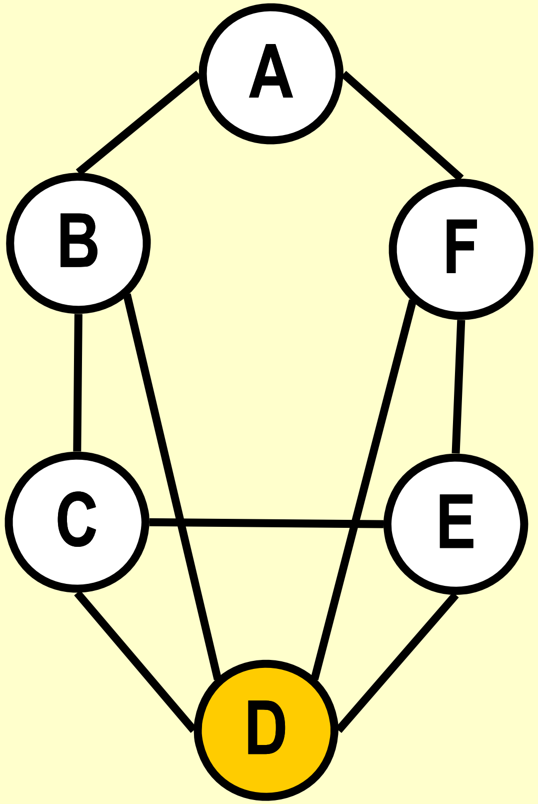
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Profundidade



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

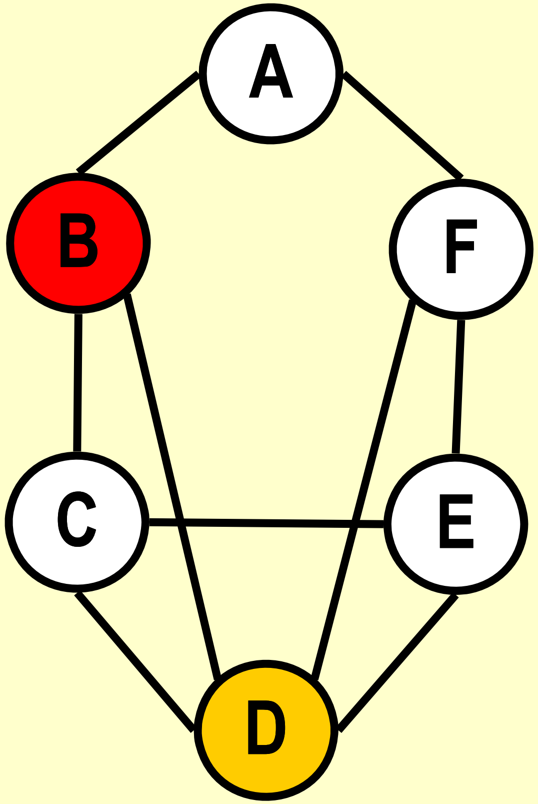
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

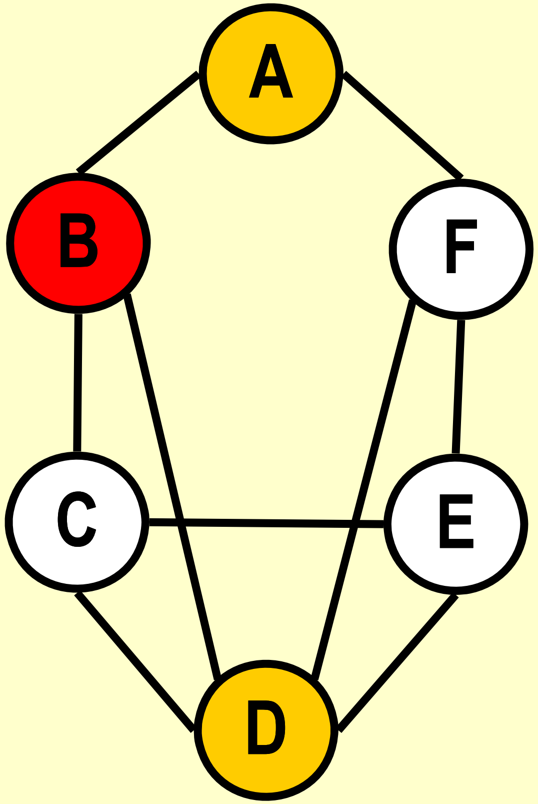
Busca em Profundidade



Adjacência:

Vértice A: [B, F]
Vértice B: [A, C, D]
Vértice C: [B, D, E]
Vértice D: [B, C, E, F]
Vértice E: [C, D, F]
Vértice F: [A, D, E]

Busca em Profundidade



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

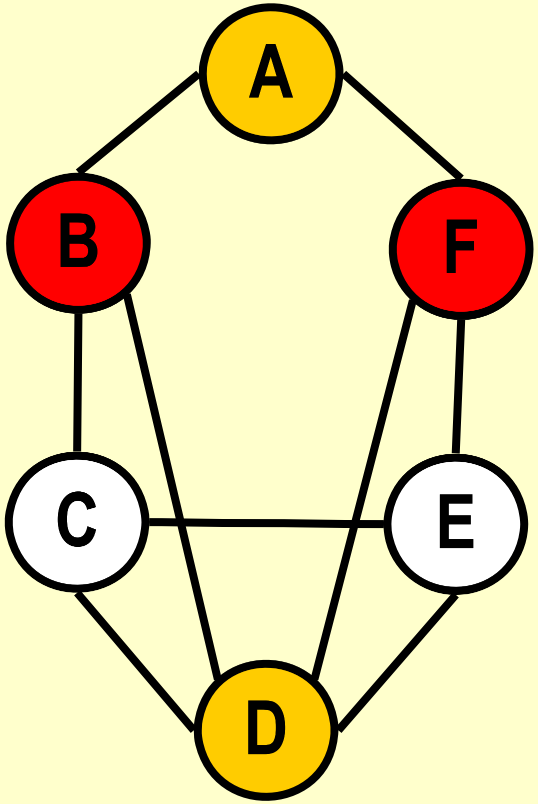
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

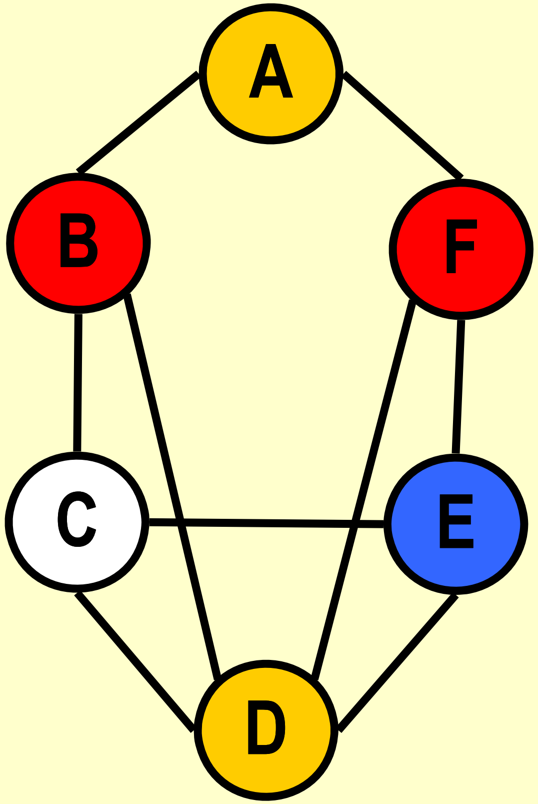
Busca em Profundidade



Adjacência:

Vértice A: [B, F]
Vértice B: [A, C, D]
Vértice C: [B, D, E]
Vértice D: [B, C, E, F]
Vértice E: [C, D, F]
Vértice F: [A, D, E]

Busca em Profundidade



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

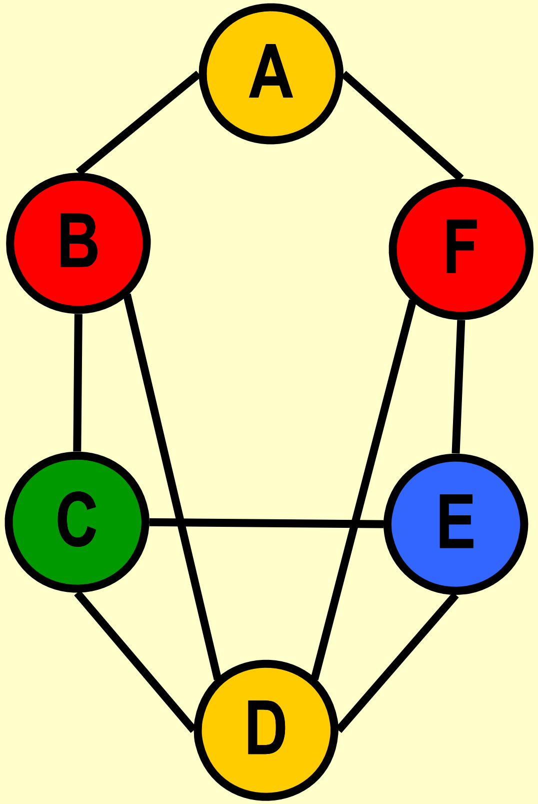
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Profundidade



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

número cromático 4

Busca em Largura

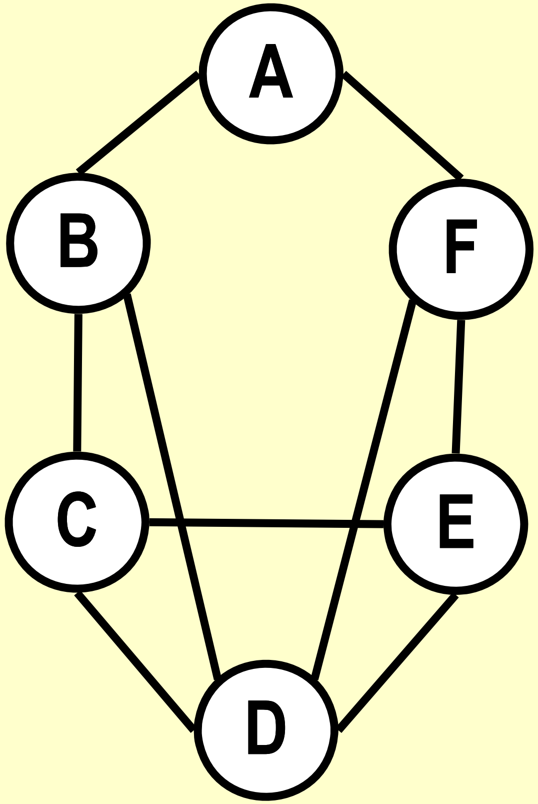
Programa Principal

```
montar a lista de adjacências
inicializar a estrutura de cores
inicializar a estrutura de fila
escolher o vertice Vi de maior grau para ser colorido primeiro
chamar a sub-rotina Colore_Vertice para colorir o vertice Vi escolhido
inserir o vertice Vi na fila Q
enquanto a fila Q nao estiver vazia faca
        remove o vertice Vk da fila
        para todo vertice Vj adjacente a Vk faça
                chamar a sub-rotina Colore_Vertice para colorir o vertice Vj
                inserir Vj na fila
        fim para
fim enquanto
```

Sub-rotina COLORE-VERTICE

```
se o vertice Vk ainda nao foi colorido
        procurar a cor C apropriada
        se nao existir cor apropriada para colorir o vertice Vk
                criar uma nova cor C
        fim se
        colorir o vertice Vk com a cor C
fim se
```

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

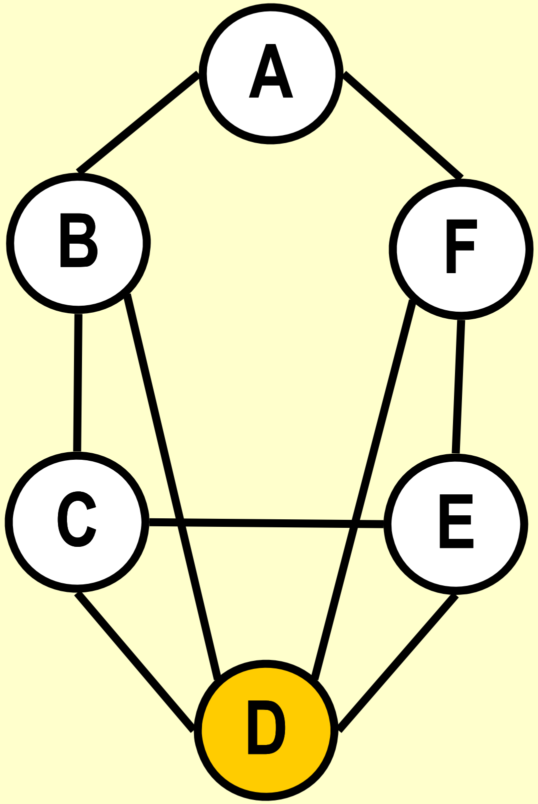
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

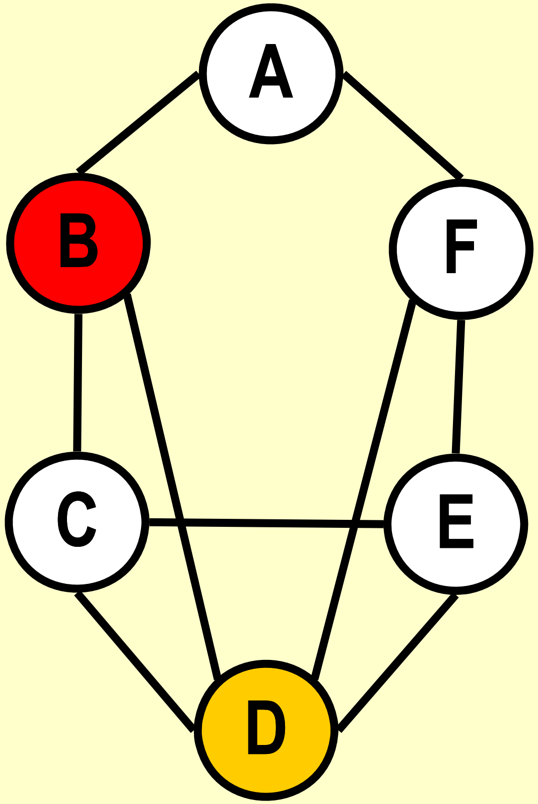
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

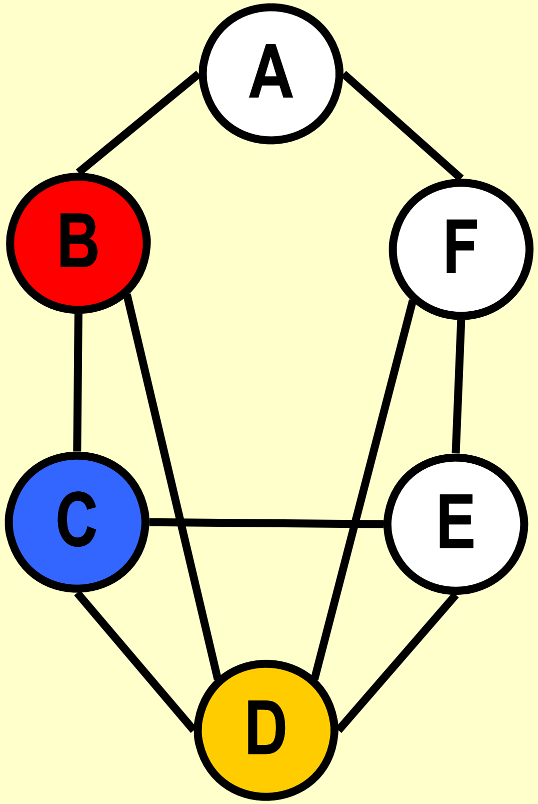
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

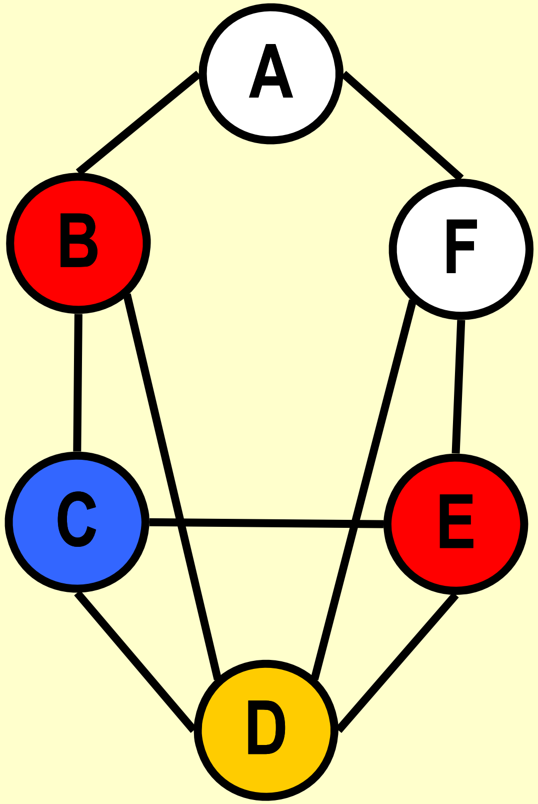
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

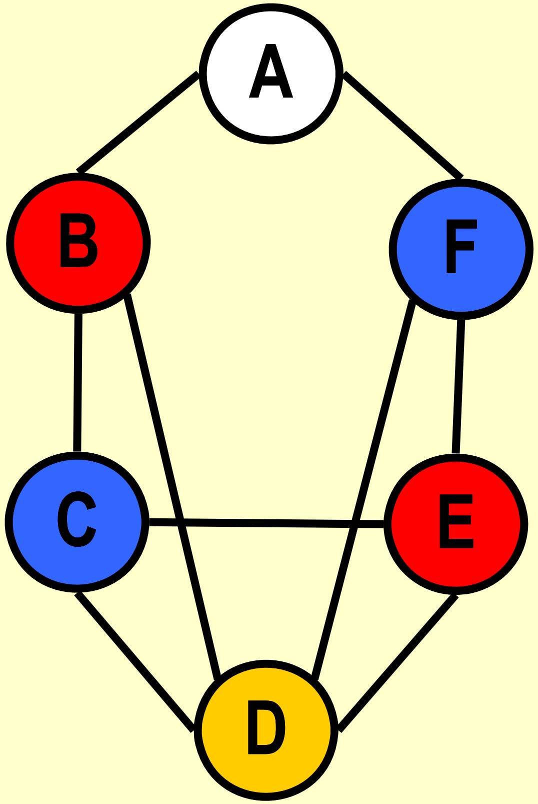
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

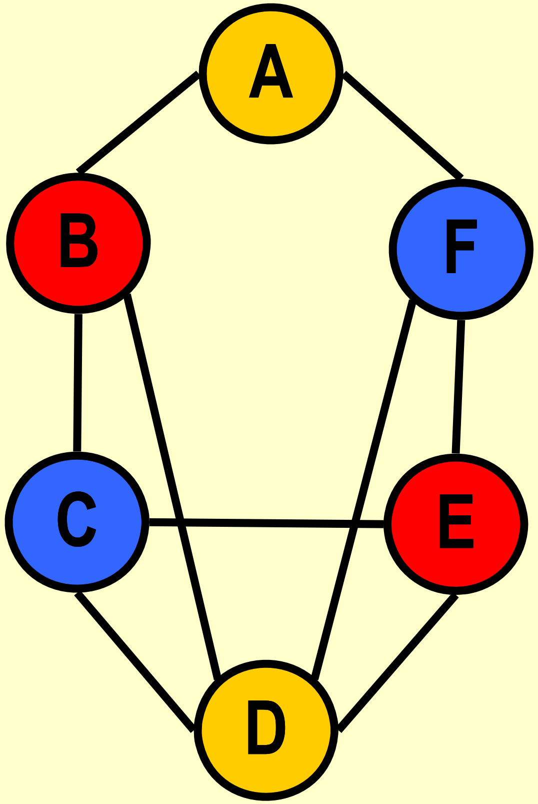
Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

Busca em Largura



Adjacência:

Vértice A: [B, F]

Vértice B: [A, C, D]

Vértice C: [B, D, E]

Vértice D: [B, C, E, F]

Vértice E: [C, D, F]

Vértice F: [A, D, E]

número cromático 3

Exercício

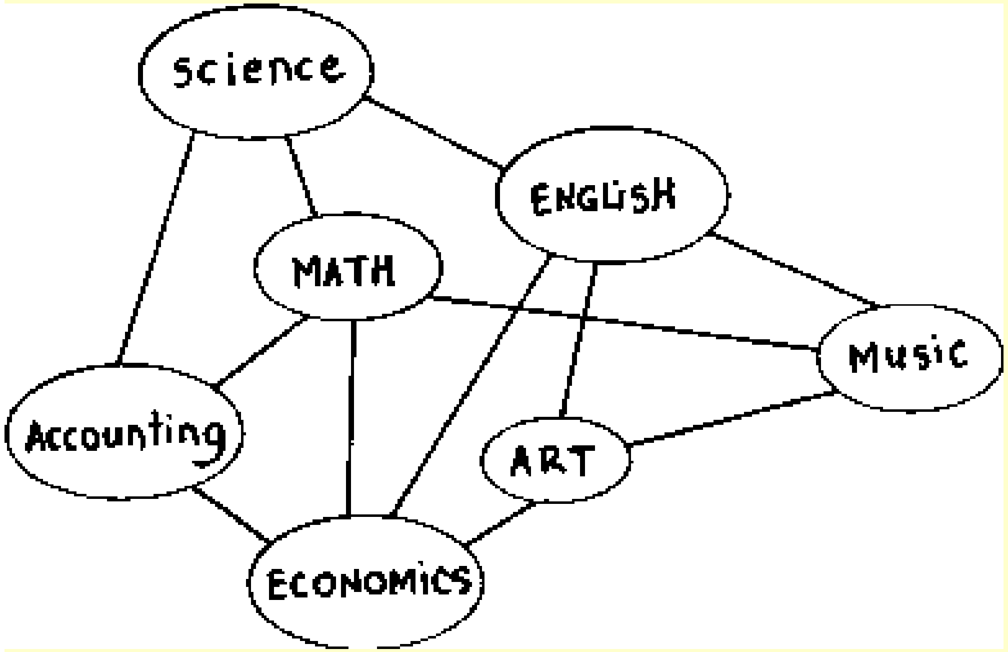
Considere a relação de exames requeridos, na mesma época, pelos alunos de uma universidade:

Código Aluno	Disciplinas
1	A, B
2	A, D
3	D, E, F
4	B, C
5	A, C
6	B, E
7	C, F
8	E, G

–Admitindo que um aluno pode executar, no máximo, 1 exame por dia, qual é o menor número de dias necessários para executar todos os exames?

1. Montar o **grafo** para a situação descrita acima, considerando:
   - os vértices representam as disciplinas oferecidas
   - uma aresta entre duas disciplina indica que pelo menos um aluno estará cursando as duas disciplinas
   - Conclusão: elas não podem acontecer no mesmo período
2. Definir o número de **cliques** do grafo
3. Definir o **CVIM** do grafo
4. Definir o **número cromático ótimo** para o grafo gerado no item 1
5. Simular o algoritmo de coloração em **largura** e em **profundidade** para verificar o número de cores gerado para o grafo.

Outro problema: como organizar um horário com o menor número possível de períodos?



- Os vértices representam disciplinas oferecidas
- Uma aresta entre duas disciplinas indica que pelo menos um aluno estará cursando ambas
- Conclusão: elas não poderão acontecer no mesmo período

Fim