1) Apresente a versão de **otimização** do Problema do Conjunto Independente de Vértices (PCIV).

R.: Dado um grafo $G = (V, E)$, obtenha um Conjunto Independente de Vértices de maior cardinalidade em G.

2) Prove que o PCIV $\in$ NP.

```
Algoritmo ConjuntoIndependente_NP
INÍCIO
    leia(A, n, k); //A: matriz de adjacência do grafo e n: número de vértices do grafo.

    para(i = 1, ..., k) faça
        c(i) ← escolha{1, ..., n}; //Escolha aleatória de k vértices para o Conj. Independente c.
    fim_para

    para(i = 1, ..., k − 1) faça
        para(j = i+1, ..., k) faça
            se(A(c(i), c(j)) = 1) então //Se o vértice da posição i é adjacente ao da posição j em c.
                retorne FRACASSO;
            fim_se
        fim_para
    fim_para

    retorne SUCESSO;
FIM
```

3) Problema da Cobertura Mínima de Vértices (PCMV): Dado um grafo $G = (V, E)$ não-orientado, dizemos que $W \subseteq V$ define uma **Cobertura de Vértices** de G se, e somente se, $\forall(i, j) \in E$, $i \in W$ ou $j \in W$.

(a) Apresente a versão de decisão do PCMV;

R.: Dado um grafo $G = (V, E)$ e $k \in \mathbb{N}^*$, existe em G uma Cobertura de Vértices de cardinalidade menor ou igual a k?

(b) Apresente a versão de otimização do PCMV;

R.: Dado um grafo $G = (V, E)$, obtenha uma Cobertura de Vértices de menor cardinalidade em G.

(c) Prove que o PCMV $\in$ NP.

```
Algoritmo Cobertura_NP
INÍCIO
    leia(A, n, k); //A: matriz de adjacência do grafo e n: número de vértices do grafo.
    cont ← 0;
    para(i = 1, ..., n) faça
        c(i) ← escolha{0,1}; //Escolha aleatória do vértice i estar (1) ou não (0) na Cobertura c.
        se(c(i) = 1) então
            cont ← cont + 1; //Obtendo a cardinalidade da Cobertura c.
        fim_se
    fim_para
    se(cont > k) então //Se a cardinalidade da Cobertura c é maior do que k.
        retorne FRACASSO;
    fim_se
    para(i = 1, ..., n − 1) faça
        para(j = i+1, ..., n) faça
            se(A(i,j) = 1) então //Se o vértice i é adjacente ao vértice j.
                se((c(i) = 0) e (c(j) = 0)) então //Se nem o vértice i e nem o j estão na Cobertura c.
                    retorne FRACASSO;
                fim_se
            fim_se
        fim_para
    fim_para
    retorne SUCESSO;
FIM
```