1) V, F, F, V, V, F, V, V, F, V

2)
```
void minMax(int *v, int ini, int fim, int *min, int *max){
    int meio, min1, max1, min2, max2;
    if (ini > fim)
        return;
    if (ini == fim)
      *min = *max = v[ini];
    else{
       meio = (ini + fim)/2;
       minMax(v, ini, meio, &min1, &max1);
       minMax(v, meio+1, fim, &min2, &max2);
       if(min1 <= min2)
          *min = min1;
       else
          *min = min2;
       if(max1 >= max2)
          *max = max1;
      else
          *max = max2;
    }
}
```

b) $$T(n) = \begin{cases} O(1) & , \text{se } n = 1 \\ 2T(n/2) + O(1) & , \text{se } n > 1 \end{cases}$$

3) PD é uma técnica p/ construir uma solução ótima baseada na composição das soluções ótimas de subproblemas, não necessariamente disjuntas. Desta forma, como a princípio as soluções dos subproblemas de todos os tamanhos menores podem ser necessárias

(indução fonte), tais soluções são computadas uma só vez e armazenadas em uma tabela, construída de forma bottom up, normalmente (mas não exclusivamente) de forma iterativa.
Força bruta testa todas as possibilidades repetindo os cálculos.

4) $a = 4,\ b = 2 \quad f(n) = n^3 \ ,\ n^{\log_b a} = n^2$

Caso 3: $f(n) = n^3 = \Omega(n^{2+1}) \quad \varepsilon = 1$

$4f(n/2) \leq cf(n) \Rightarrow 4\frac{n^3}{8} \leq cn^3 \quad \frac{n^3}{2} \leq cn^3$

vdd p/ $c = \frac{1}{2}$

$T(n) = \theta(n^3)$