S T Q Q S S D ___/___/___

José Douglas Jardim Soares, 485347

3-

```
Algoritmo Kesimo(X[b..n], Y[c..M], int K){
  Se(b = n) retorna Y[c + K - 1];
  Se(c = M) retorna X[b + K - 1];
  Se(K = 0 ou K > (n-b) + (M-c)) retorna "K inválido";

  Se(K = 1){
     Se(X[b] < Y[c]) retorna X[b];
     Senão retorna Y[c];

  int atual ← K/2;
  Se(atual - 1 >= n - b){
     Se(X[n-1] < Y[c + atual - 1])
        retorna Y[c + (K - (n-b) - 1)];
     Senão
        retorna Kesimo(X[b..n], Y[(c+atual)..M], K-atual);
  }
  Se(atual - 1 >= M - c){
     Se(Y[M-1] < X[b + atual - 1])
        retorna X[b + (K - (M-c) - 1)];
     Senão
        retorna Kesimo(X[b+atual..n], Y[c..M], K-atual);
  }Senão{
     Se(X[atual + b - 1] < Y[atual + c - 1])
        retorna Kesimo(X[b + atual..n], Y[c..M], K-atual);
     Senão
        retorna Kesimo(X[b..n], Y[c+atual..M], K-atual);
  }
```

S T Q Q S S D __/__/__

AP1 - CANA José Douglas Gondim Soares, 485347

1-) a)

i		k						j
15	42	21	50	33	65	40	43	20
1	2	3	4	5	6	7	8	9

$i = 1, j = 9, k = 3$

1º L6

i	k				j			
15	42	21	50	33	65	40	43	20
1	2	3	4	5	6			

$i = 1, j = 6, k = 2$

2º L6

i,k			j					
15	42	21	50	33	65	40	43	20
1	2	3	4					

$i = 1, j = 4, k = 1$

3º L6

i,k		j						
15	42	21	50	33	65	40	43	20
1	2	3						

$i = 1, j = 3, k = 1$

4º L6

i	j							
15	42	21	50	33	65	40	43	20
1	2							

$i = 1, j = 2, k = 0$

3º L6, 1º L7

k	i	j						
15	21	42	50	33	65	40	43	20
1	2	3						

$i = 2, j = 3, k = 0$

3º L6, 1º L7, 1º L8

k	i	j						
15	21	42	50	33	65	40	43	20
1	2	3						

$i = 2, j = 3, k = 0$

retorno ao estado 2º L6 com

15	21	42	50	33	65	40	43	20

$i = 1, j = 4, k = 1$

2º L6 1º L7

	i		j					
15	21	42	50	33	65	40	43	20
1	2	3	4					

$i = 2, j = 4, k = 1$

2º L6 1º L7 1º L8

	i	j						
15	21	42	50	33	65	40	43	20
1	2	3						

$i = 2, j = 3, k = 0$

CONTINUA

S T Q Q S S D

__/__/__

retorna ao 1º L6 com | 15 | 21 | 42 | 50 | 33 | 65 | 40 | 43 | 20 | $i=1, j=6, k=2$

1º L6 1º L7 | 15 | 21 | 42 | 50 | 33 | 65 | 40 | 43 | 20 | $i=3, j=6, k=1$

1 2 3 4 5 6

(b) O algoritmo vai ordenar o vetor sempre de 3 em 3 posições.

(c)

S T Q Q S S D   __/__/__

José Douglas Gardim Soares, 485347

2- O algoritmo funciona usando a partição para procurar pelo elemento central $n/2$. Se a condição da linha 1 for verdade, o array só tem 1 elemento que já vai ser a mediana do array original, então podemos retorná-lo.
O algoritmo está buscando pelo elemento $n/2^{th}$, o que se assemelha ao problema da seleção. Se eu particionei o array e ainda não estou na posição $n/2$, se a posição que estou for menor, procure particionar o array à direita em busca da posição $n/2$. Senão, procura à esquerda.

Caso base: $p = r = 1$
A linha 2 retornará o único valor do array, que já é a mediana. Correto

Assumindo que é correto para $K < n$

Digamos K_{esq} ser o subarray esquerdo e K_{dir} o subarray direito após a partição da linha 4.
Podemos afirmar que ambos estão ordenados, pois partimos do pressuposto que a partição funciona corretamente, também afirmamos que o pivot foi colocado na sua posição correta se o array estivesse ordenado.
Logo, podemos saber a cada partição se o elemento que pertence à posição $n/2$ do array ordenado (a mediana) é o elemento que usamos como pivot ou se devemos procurar à direita ou à esquerda dele.

CONTINUA...

__/__/__ S T Q Q S S D

O algoritmo só vai parar quando ele escolher o elemento que pertence a posição n/2 da array ordenada como pivot ou se só restou 1 elemento no array e ele já vai ser a mediana da array original.