Assuma um sistema com os tipos de recursos (A, B,…), processos (P1, P2,…) e caracterização como apresentada nas tabelas seguintes:

Alocado

	A	B	C
P0	0	0	2
P1	2	4	1
P2	3	0	2
P3	2	0	0

Necessidades máximas

	A	B	C
P0	4	3	3
P1	2	6	2
P2	9	0	2
P3	3	2	2

Disponibilidade

A	B	C	
2	3	1	

Aplique o algoritmo do banqueiro e determine se existe uma sequência de execução que mantenha o sistema num estado seguro. Apresente esta sequência juntamente com os cálculos que sentir necessidade de efetuar.

Proposta de solução:

Recorda-se que o algoritmo do banqueiro aplica-se, no contexto dos sistemas operativos, na gestão de recursos computacionais (num cenário de múltiplas instâncias por tipo de recurso) bem como um "mecanismo" para evitar bloqueios (*deadlocks*). No exemplo acima podemos verificar existem três tabelas "Alocado", "Necessidades Máximas" e "Disponibilidade".

Para obter a "necessidade", ou seja, o pedido de recursos, que é efetuado por cada processo neste instante de tempo? Obtém-se aplicando subtraindo as necessidades máximas ao alocado. Assim, temos:

Request(P0) = 431
Request(P1) = 21
Request (P2) = 600
Request (P3) = 122

A aplicação do algoritmo passará por verificar para cada de processo na fila se Request(Px) <= Available. Se sim, atualiza-se a disponibilidade (disponibilidade = disponibilidade + alocado) e o processo poderá terminar. Caso contrário, avança-se para o próximo processo na fila.

E.g., Request(P0) <= Avalibable ⇔ 431 <= 231 -> o que é falso. Passamos para o segundo processo na fila: Request(P1) <= Avalibable ⇔ 21 <= 231 -> o que é verdade. Logo P1 poderá terminar.

Assim, neste exemplo, teremos a seguinte sequência de disponibilidades:

Originando a seguinte sequência de execução: <P1,P3,P0,P2>. Como todos os processos terminaram então o sistema encontra-se num estado seguro.