

## Código de acesso

Com o intuito de evitar que alguns alunos utilizem o LabGrad para jogar Counter-Strike durante toda madrugada, (quem nunca?!) foi implementado um controle de acesso digital, onde somente o monitor possui a tão desejada senha de acesso, que deve ser digitada toda manhã para a abertura do laboratório.

Sabe-se que a tecnologia utilizada para implementar a leitura da senha no teclado pinpad utiliza internamente um buffer em formato de fila e que, assim que a senha correta for digitada, independente do que foi digitado anteriormente, a porta se abrirá.

Com o passar do tempo, foi possível notar que algumas teclas (comumente utilizadas) começaram a apresentar desgaste, possibilitando aos famigerados viciados em Counter-Strike tentar adivinhar a tão desejada senha de acesso. Para exemplificar, a figura abaixo apresenta como poderia estar o teclado numérico da porta, onde as teclas 1, 3 e 7 estão claramente desgastadas.



Figura 1: Exemplo do teclado

Testar todas as possíveis combinações pode ser tornar uma tarefa árdua, mas um dos gênios da computação sugeriu que, dado as teclas falhas, juntamente com o tamanho a senha conhecida (pela documentação do fabricante) e o fato da tranca implementar um buffer em formato de fila; é possível implementar um algoritmo que forneça a sequência de teclas a ser pressionada que contenha todas as possíveis combinações de senha, considerando a sobreposição das teclas na sequência.

Para o teclado acima, a seguinte sequência contém todas as possíveis combinações de senhas de 4 dígitos que contenham os dígitos 1, 3 e 7:

1 1 1 1 3 1 1 1 7 1 1 3 3 1 1 3 7 1 1 7 3 1 1 7 7 1 3 1 3 1  
7 1 3 3 3 1 3 3 7 1 3 7 3 1 3 7 7 1 7 1 7 3 3 1 7 3 7 1 7 7  
3 1 7 7 7 3 3 3 3 7 3 3 7 7 3 7 3 7 7 7 7 1 1 1

Figura 2: Todas as possíveis combinações com 1, 3 e 7

Um outro exemplo é o seguinte trecho (incompleto) que cobre 10 possíveis combinações de senhas contendo os dígitos 1, 2, 3 e 4, reduzindo o tamanho de teclas pressionadas de 40 (10x4) para 29:

1	1	1	1	2	1	1	1	3	1	1	1	4	1	1	2	2	1	1	2	3	1	1	2	4	1	1	3	2
1	1	1	1																									
	1	1	1	2																								
					1	1	1	3																				
									1	1	1	4																
		1	1	2	1								1	1	2	2												
																	1	1	2	3								
																					1	1	2	4				
						1	1	3	1																1	1	3	2

Figura 3: Exemplo com 10 combinações

Nota-se que não necessariamente uma tecla desgastada fará parte da senha, dessa forma, uma senha de 2 dígitos que possua as teclas 3 e 4 pode ser identificada pela sequência 33443, de forma que todas as combinações (33, 34, 43 e 44) sejam cobertas.

Seu trabalho é implementar um algoritmo que, dado o tamanho da senha e as teclas visivelmente desgastadas, forneça a menor sequência (numérica e em tamanho) de teclas que, se pressionadas, cobriria todas as possíveis combinações de senhas para a abertura da porta.

### Entrada

A entrada consiste em vários casos de testes. A primeira linha de um caso de teste contém dois inteiros  $N$  e  $K$  ( $0 \leq N, K \leq 8$ ), lidos da entrada padrão, representando, respectivamente, o tamanho da senha do teclado numérico (visto no manual) e o número de teclas visivelmente desgastadas.

A linha seguinte contém  $K$  dígitos  $D_i$  ( $1 \leq i \leq k$ ), representando em ordem crescente as teclas numéricas que possuem desgaste visível. Quando  $N=0$  e  $K=0$  deve-se considerar o fim da execução do programa.

### Saída

Para cada caso de teste, seu programa deve imprimir uma única linha contendo a menor sequência numérica de dígitos a ser pressionado (sem espaços), que cubra todas as possíveis combinações de senha, dado o tamanho de senha  $N$  e as  $K$  teclas fornecidas.

### Exemplo

Entrada	Saída
2 2 3 4	33443 4454655664
2 3 4 5 6	555565557556655675576557756565756665667567 6567757576657677765777666676677676777555
4 3 5 6 7	111131117113311371173117713131713331337137 313771717331737177317773333733773737777111
4 3 1 3 7	
0 0	

