RA	Comentários		Não entregou	Não passou nos testes	Indentação inconsistente ou errada	Não utiliza fila de prioridades mínima implementada como heap	Implementa a fila de prioridades mínima de maneira incorreta	Não respeita o tempo esperado de O (M log(n)) do algoritmo.	Notas parciais	Fora do prazo	Nota final
		Penalidade (pontos)	10	10	0	10	1	1		0.75	
G											
135526									10		10
138733			x						0	x	0
147084			x						0		0
150990			x						0		0
156539			x						0		0
156719									10		10
168081			x						0		0
168684			x						0		0
168738			x						0		0
169492			x						0		0
169886	O heap tem um vetor de ponteiros pra No. Não tem uma função para construir o heap, e constroi na main quando le o n.								10		10
171395	Não implementa explicitamente um TAD fila de prioridades/heap. Insere cada vetor interno em um vetor externo, e heapifica esse vetor externo.								10		10
172043			x						0		0
173656									10	x	7.5
174308									10	x	7.5
176293									10		10
176746			x						0		0
178685	Quando adiciona um elemento ao vetor interno (com lista encadeada) adiciona sempre ao fim do vetor, levando a uma leitura O(n^2).							x	9	x	6.75
182284	A função cria heap já insere cada vetor interno e já vai fazendo downheap.								10		10
182532									10		10
184336			x						0		0
186130			x						0		0
186586	Não separa bem a implementação do algoritmo, atualizando a chave de cada vetor interno e a quantidade de vetores internos apenas no cliente (lab06.c)								10	x	7.5
188171			x						0		0
202495			x						0		0
210792			x						0		0
H											

123767	Utiliza uma função iterativa conserta_heap que atua como ambos upheap e downheap. Citou adaptação de um algoritmos dos slides de um prof. Francescini do IC. Não implementa explicitamente um TAD fila de prioridades/heap. Usa apenas uma struct para o vetor interno.								10		10
138596			x						0		0
141692			x						0		0
157510			x						0		0
159679	Criou uma função muda_prioridade que só altera a chave do vetor no topo do heap e faz upheap/downheap. Até implementou uma função extrai_mínimo, mas só usa ela quando o vetor interno tem tamanho 1, quando ele remove mesmo o vetor.								10		10
160004	Remove os números do vetor interno, fazendo shift do restante dos elementos do vetor para a esquerda							x	9	x	6.75
165685									10		10
166527									10	x	7.5
168342									10	x	7.5
170553			x						0		0
171859			x						0		0
172910									10		10
172967			x						0		0
173033			x						0		0
174217	Remove os números do vetor interno, fazendo shift do restante dos elementos do vetor para a esquerda. O (M^2log(n)).							x	9		9
175586			x						0		0
175717	Chama um heapify por inserção/remoção no heap. Encontra o menor elemento dos vetores internos linearmente antes de remover o vetor interno do heap. Fez um vetor de heaps, e não um heap de vetores.						x	x	8		8
176573			x						0		0
178369			x						0		0
178663	Remove os números do vetor interno, fazendo shift do restante dos elementos do vetor para a esquerda. Em cada caso de extração do menor, realiza um sobe_no_heap e um desce_no_heap							x	9	x	6.75
182371									10		10
182509			x						0		0
182762			x						0		0
183266									10		10
184083									10		10
186291	Warning de compilação: unused-value.								10	x	7.5
188718									10		10