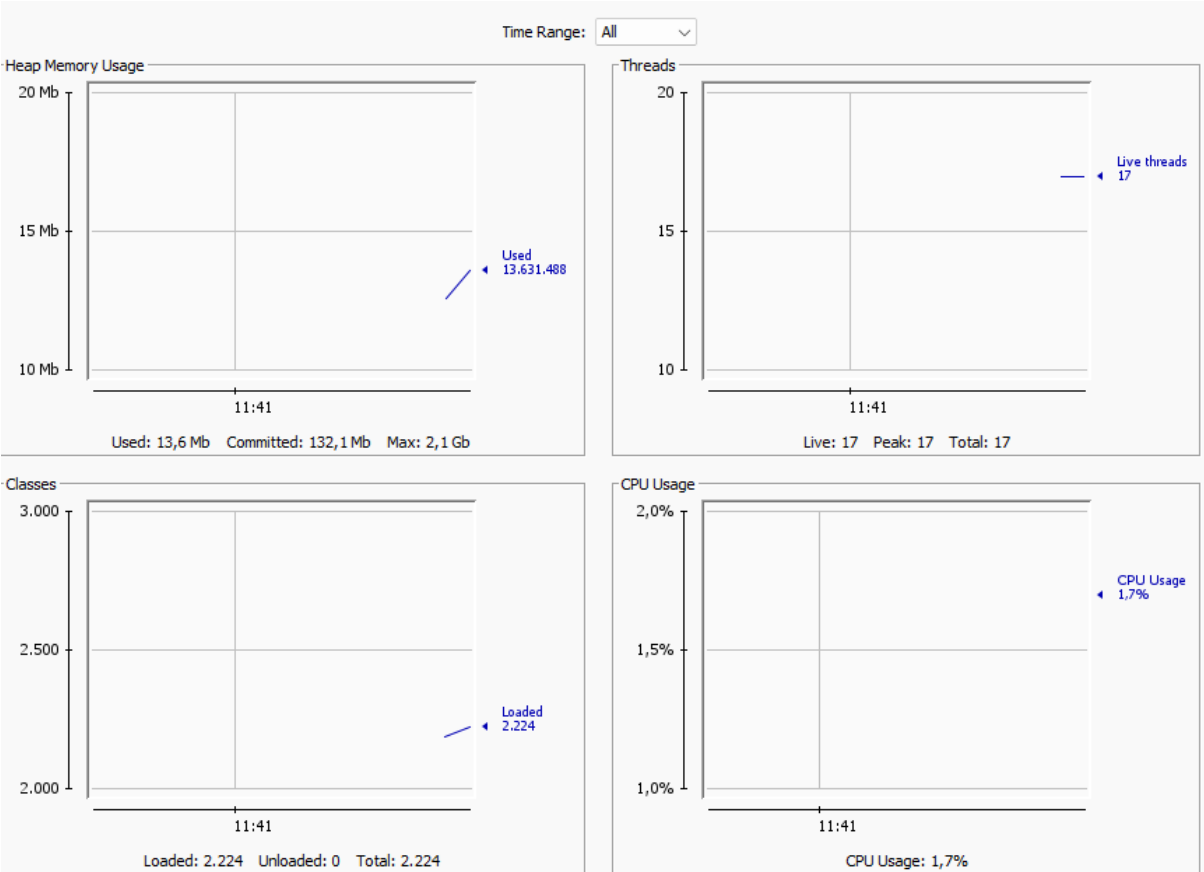


Relatório árvore de busca e AVL

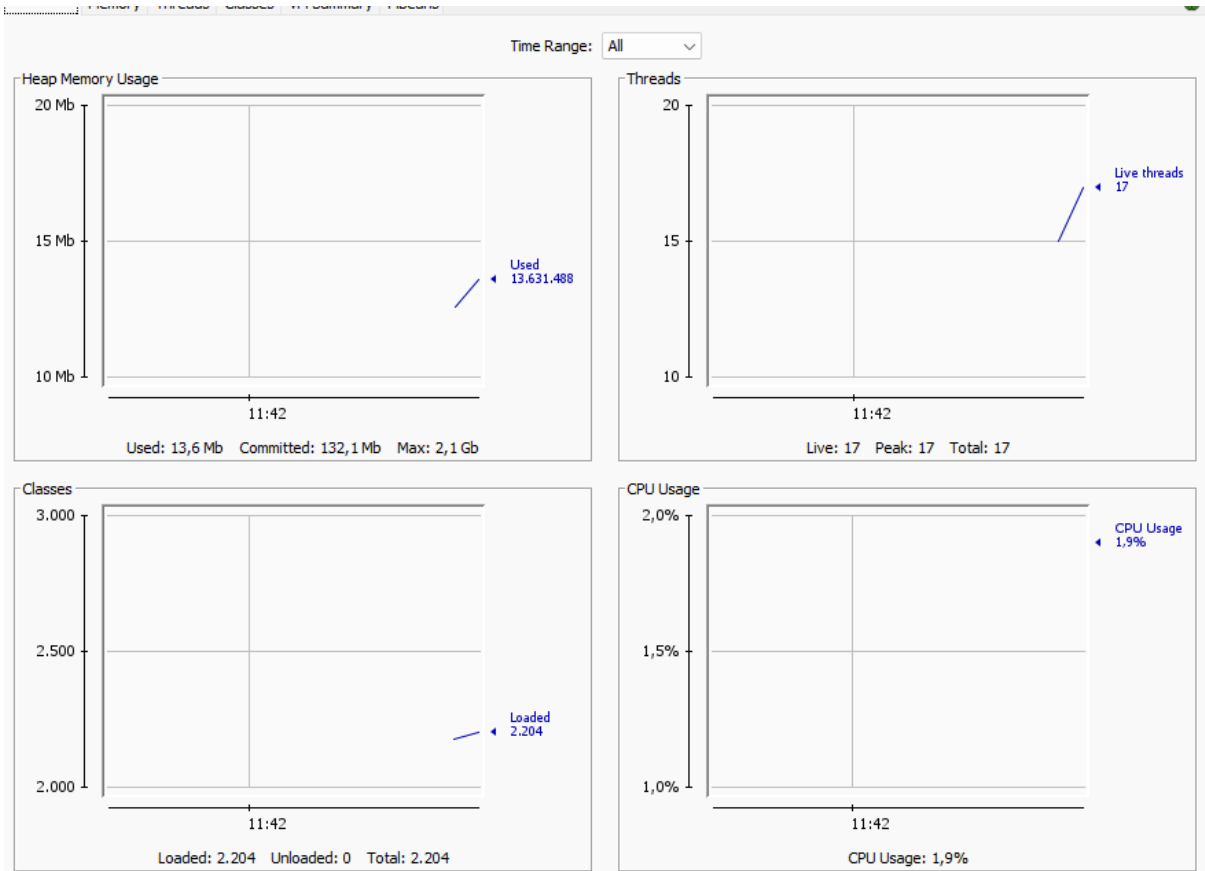
Estudante: João Pedro Igeski Moraes

Gráficos

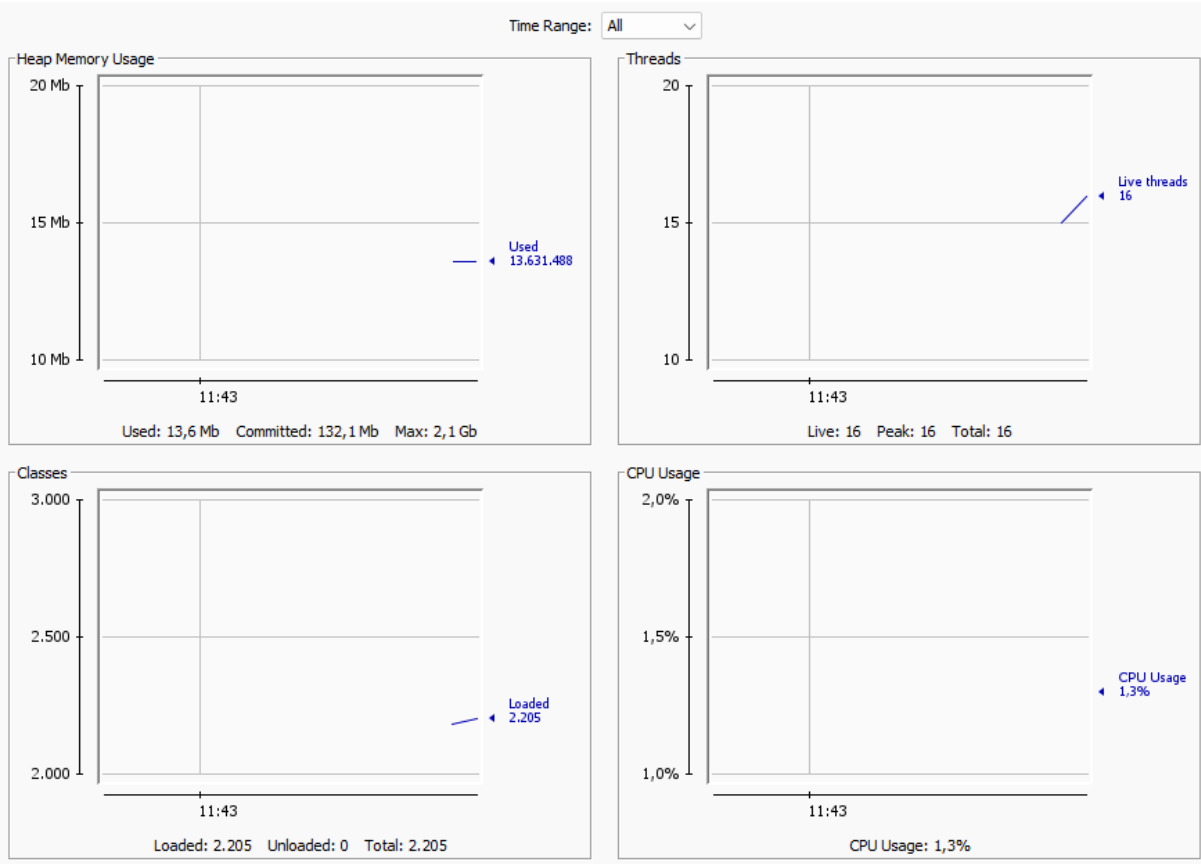
Inserir 100 (árvore de busca)



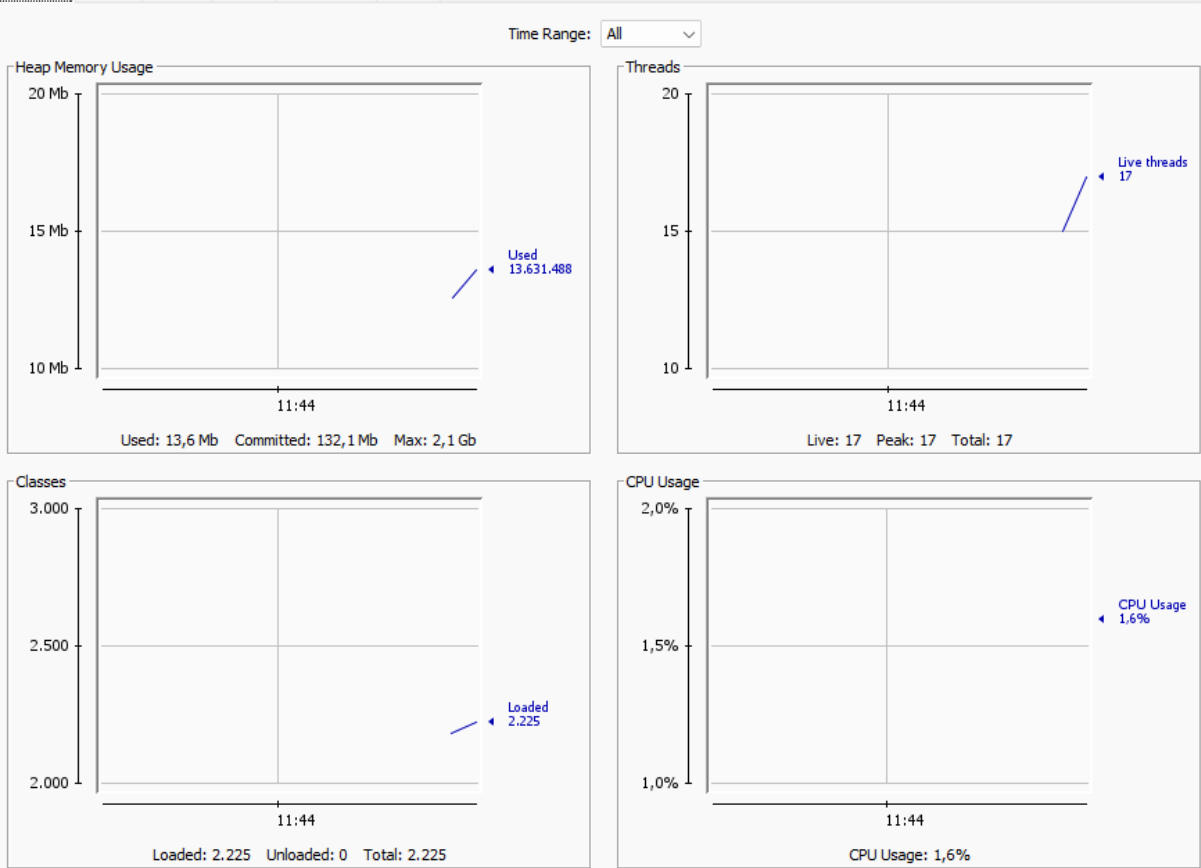
Inserir 500 (Busca)



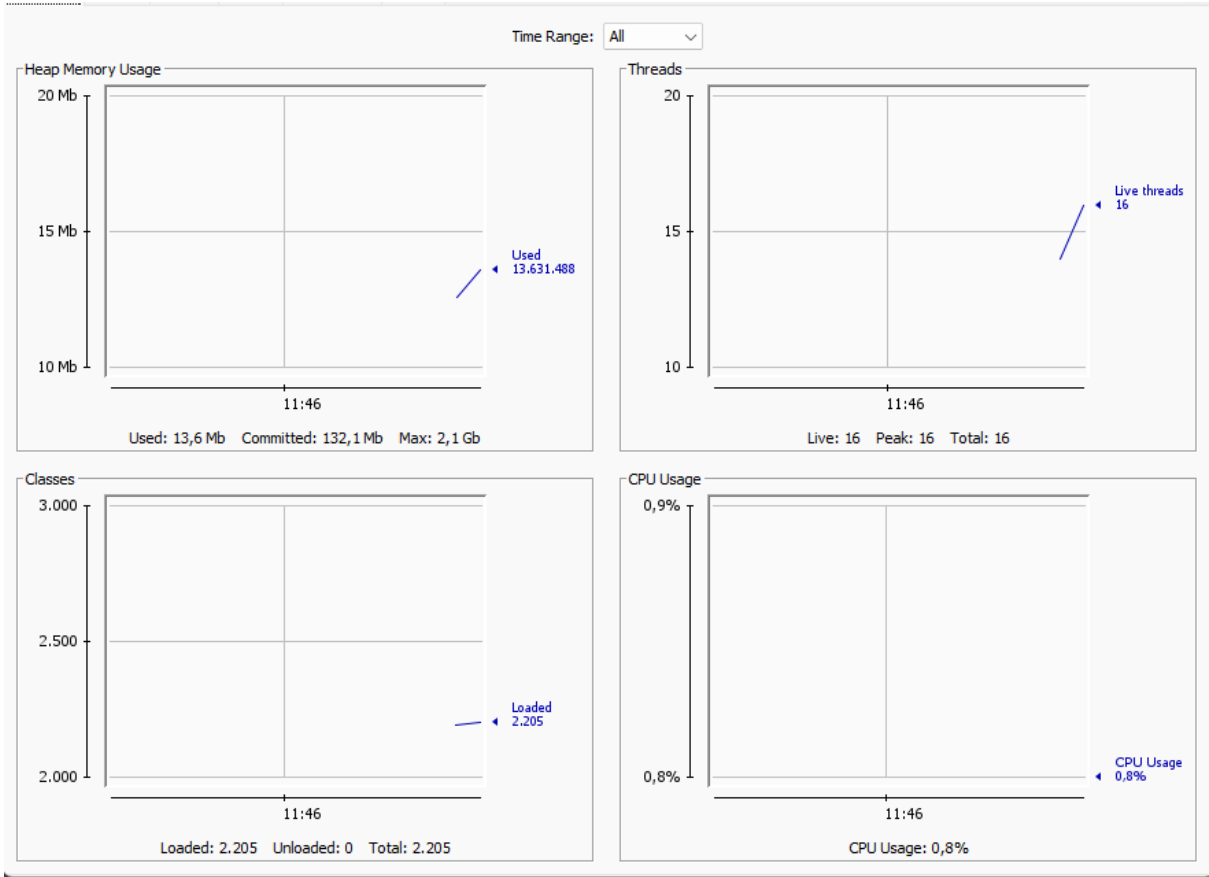
Inserir 1000 (Busca)



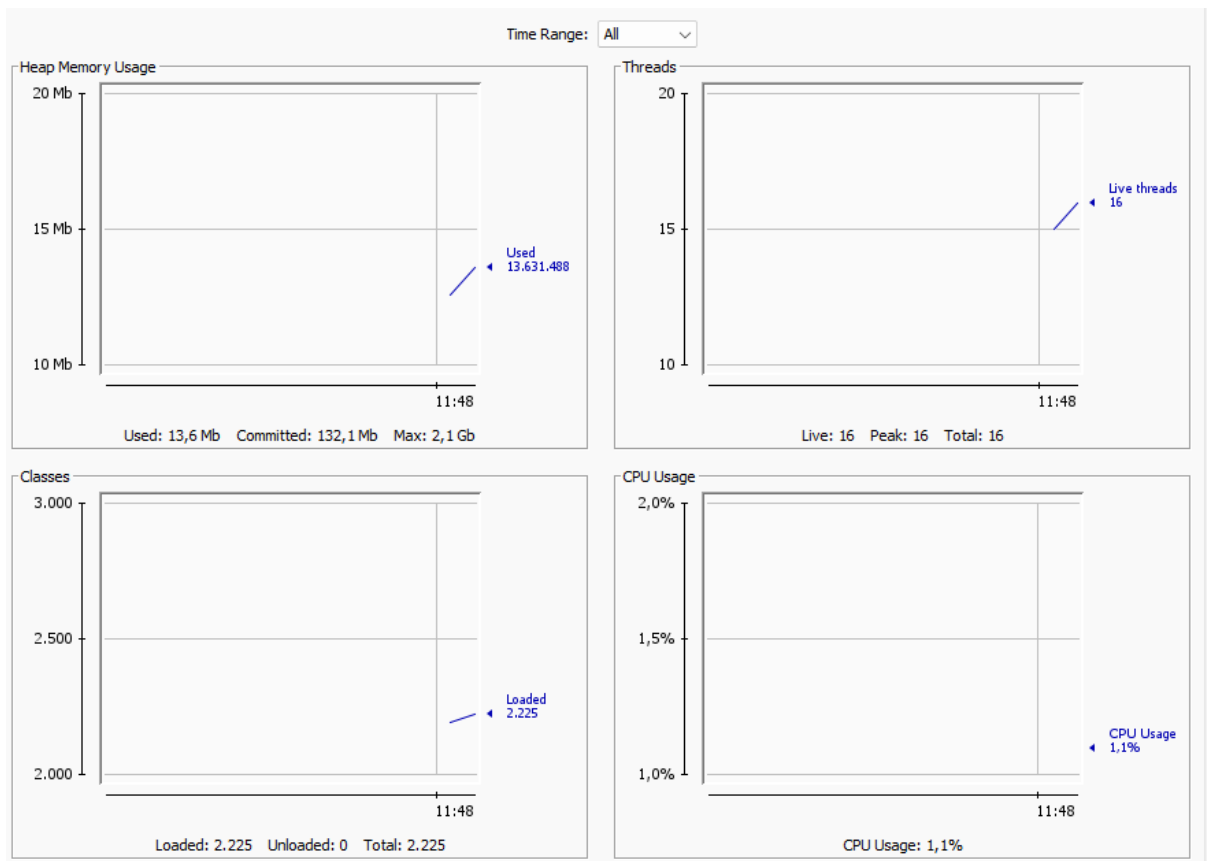
Inserir 10000(Busca)



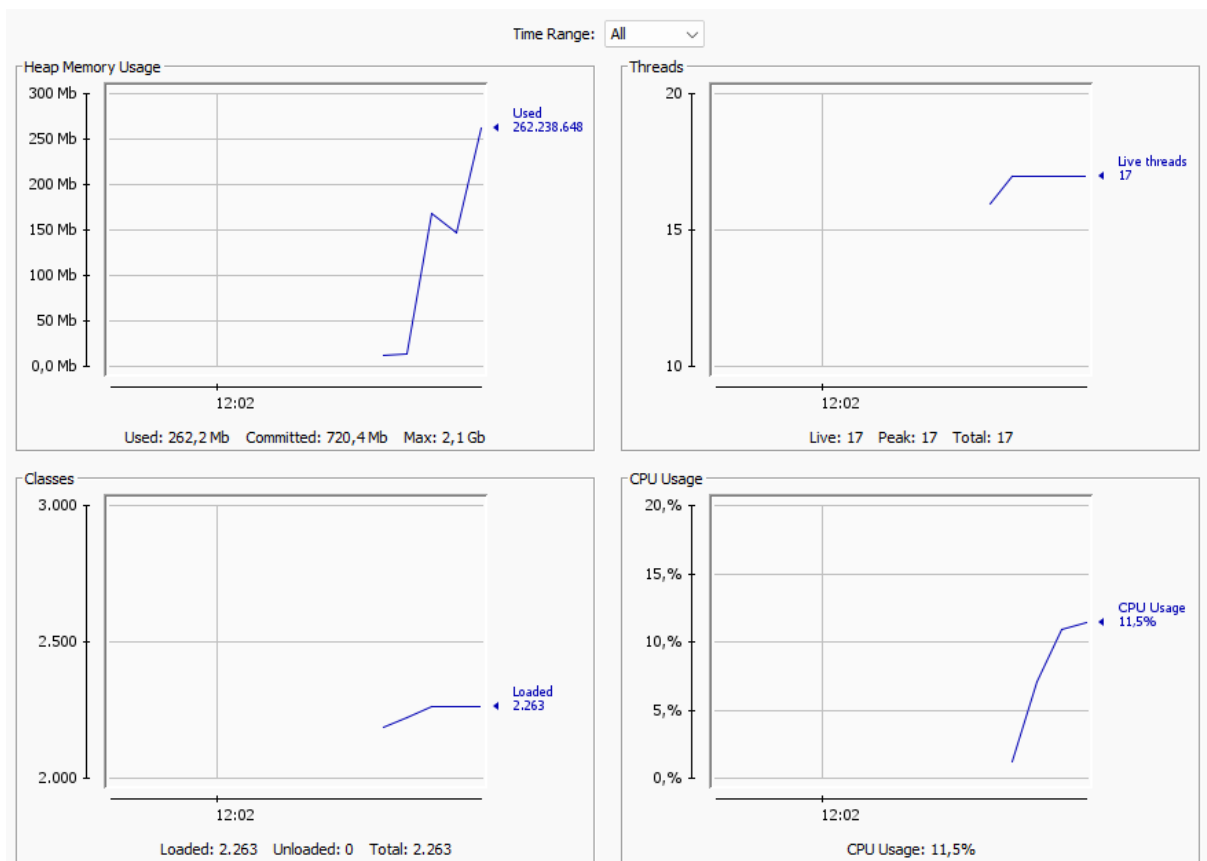
Inserir 20000 (Busca)



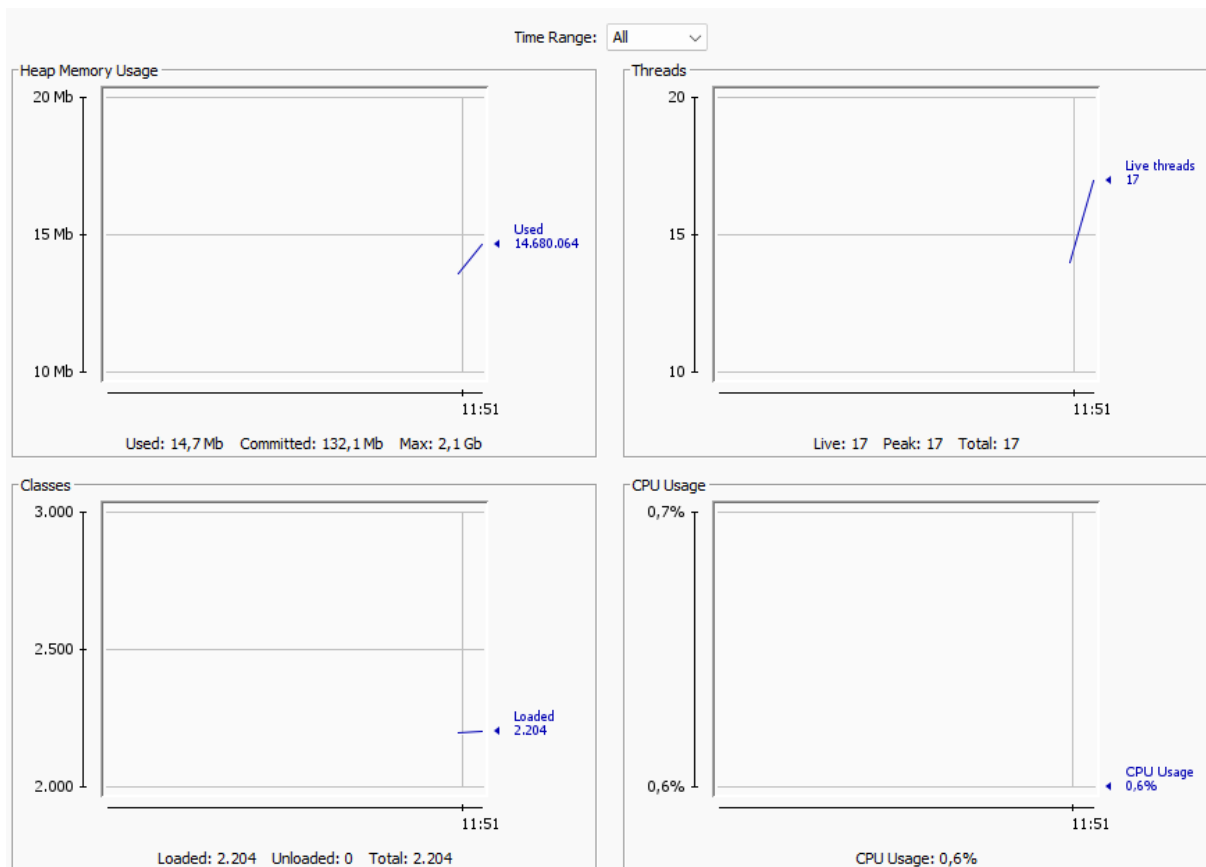
Inserir 100000 (Busca)



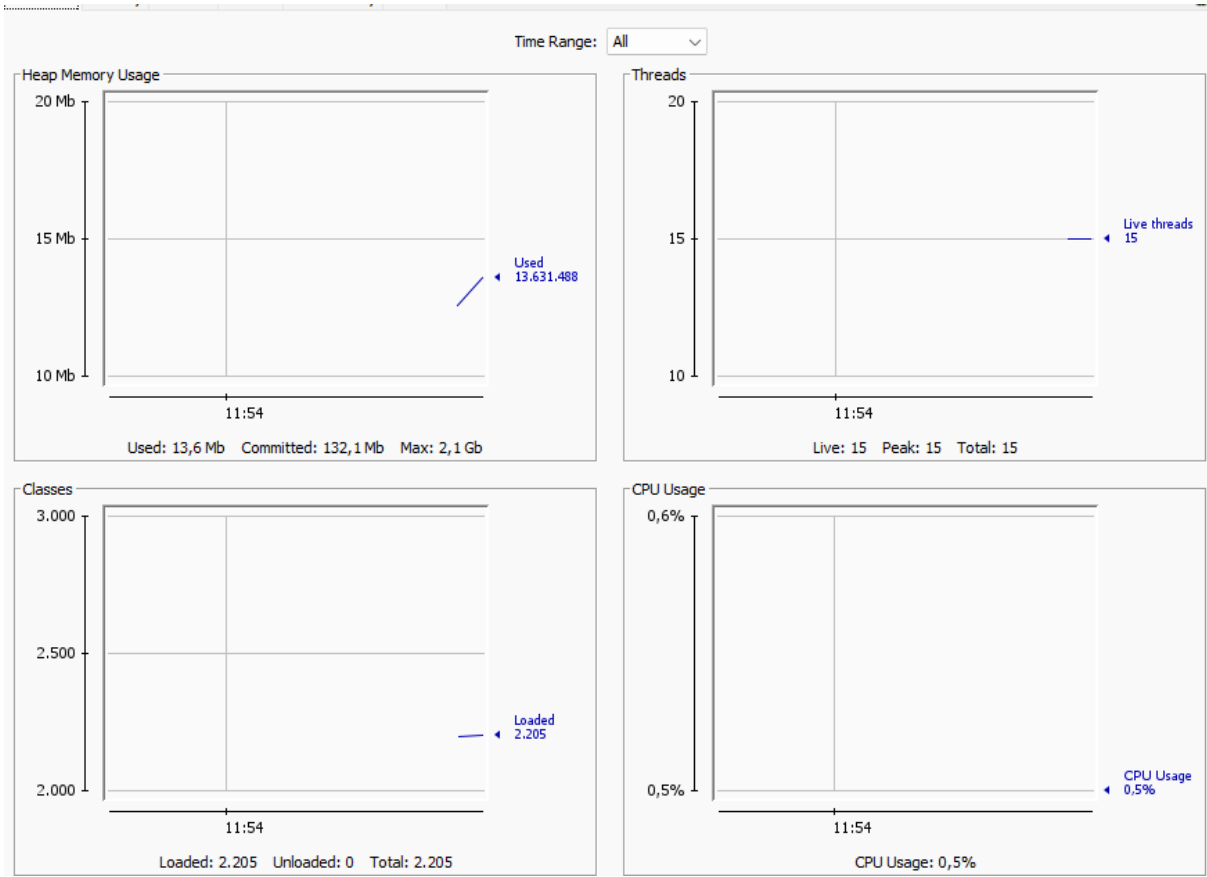
Inserir 1000000 Busca



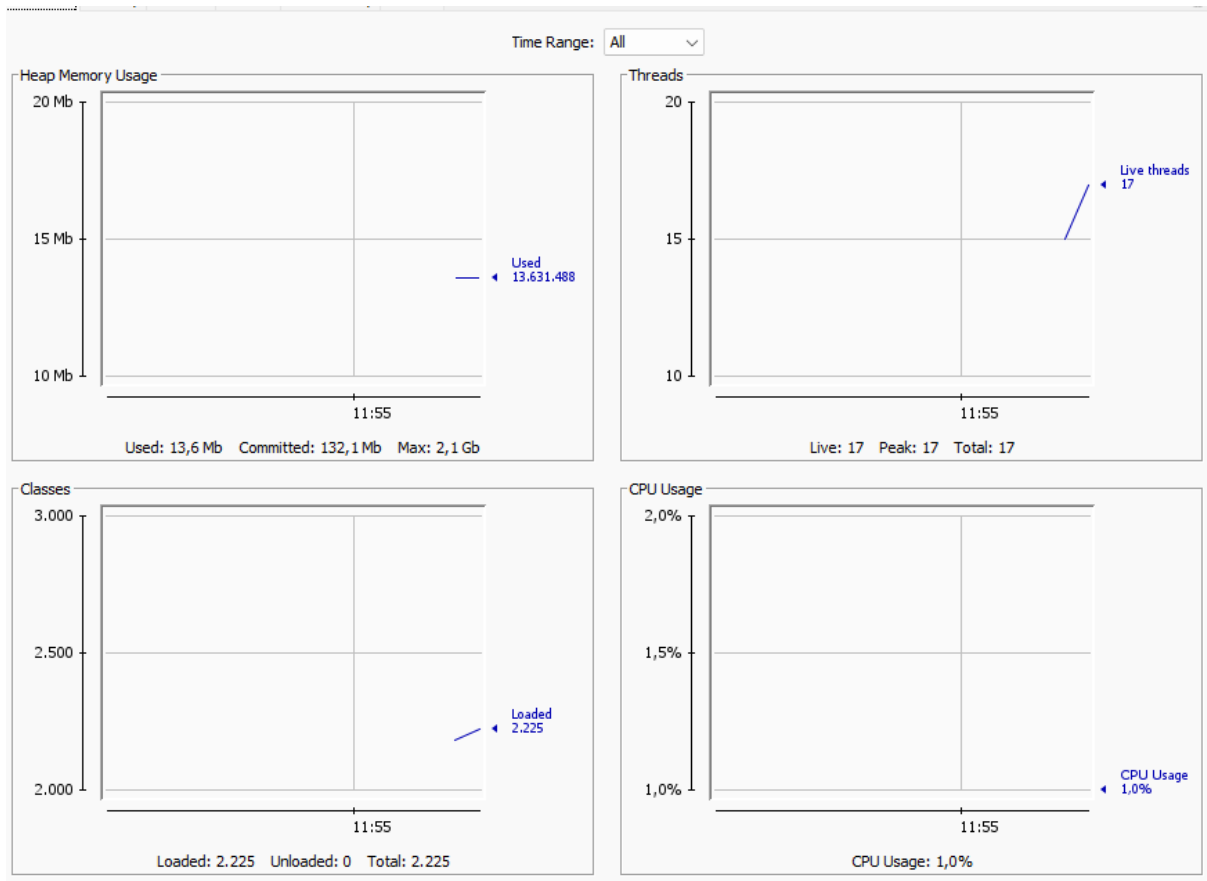
Inserir 100000 AVL



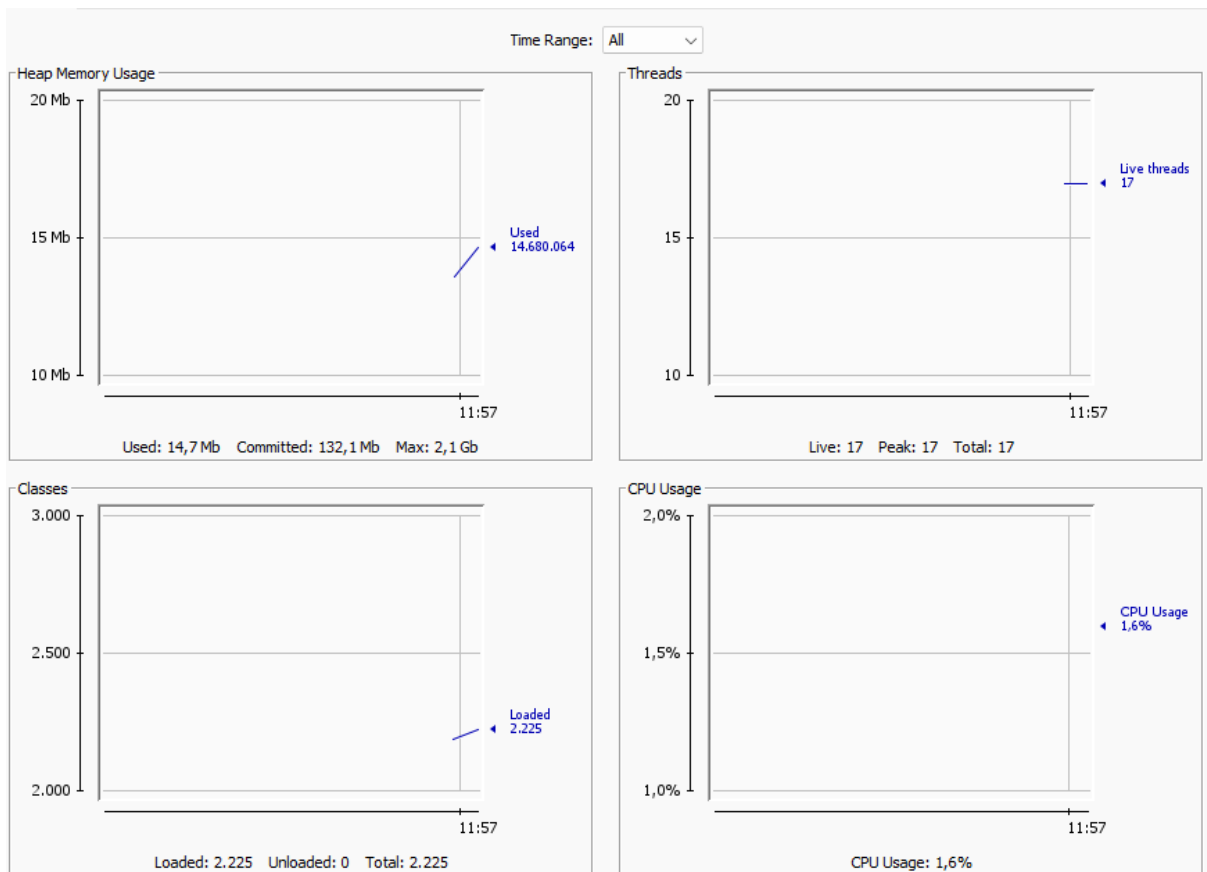
Inserir 100 AVL



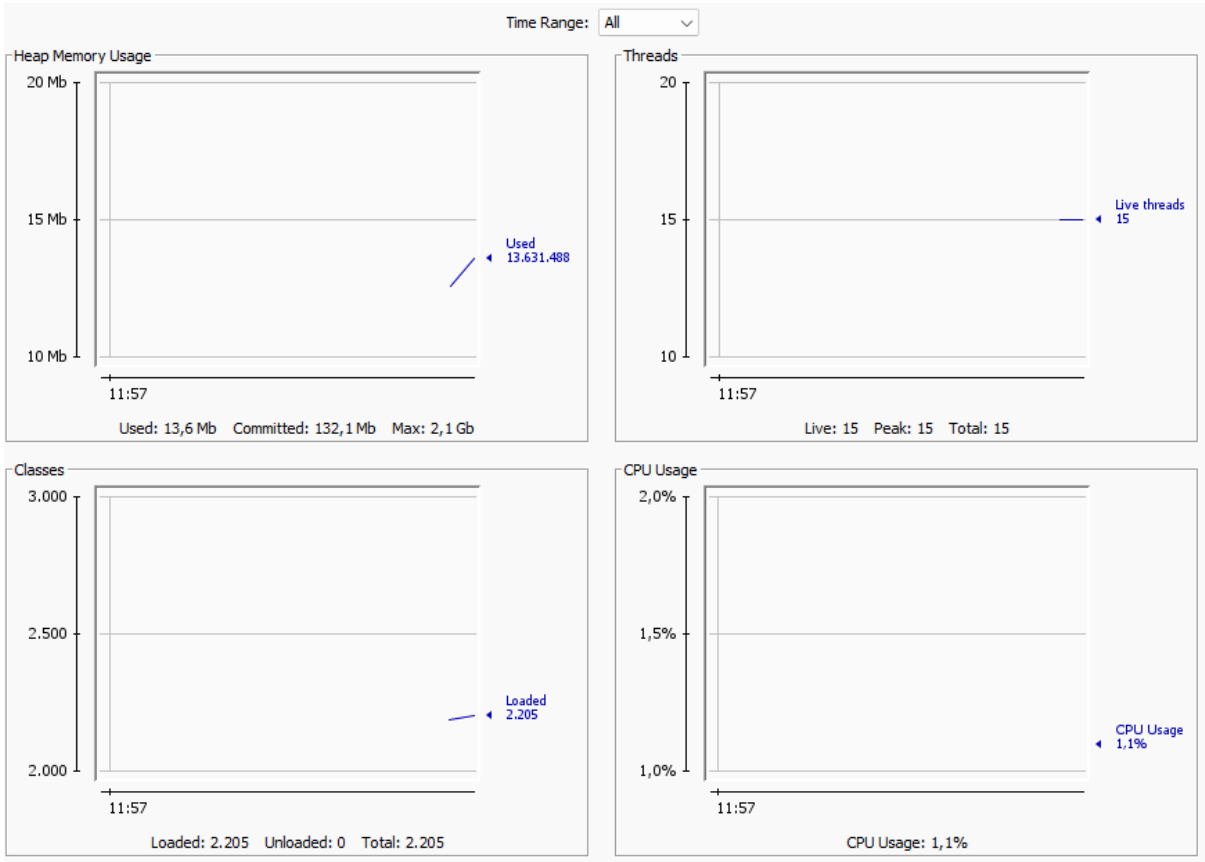
Inserir 500 AVL



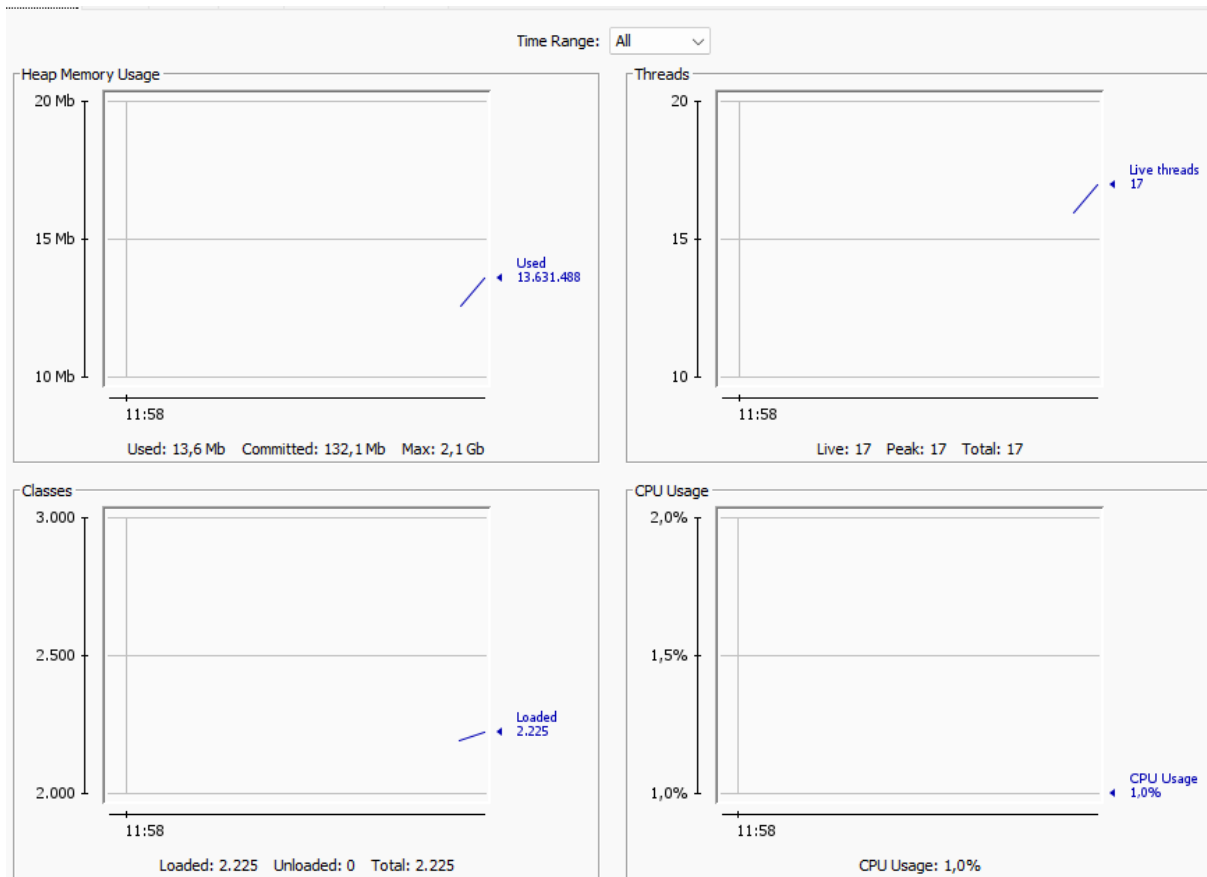
Inserir 1000 AVL



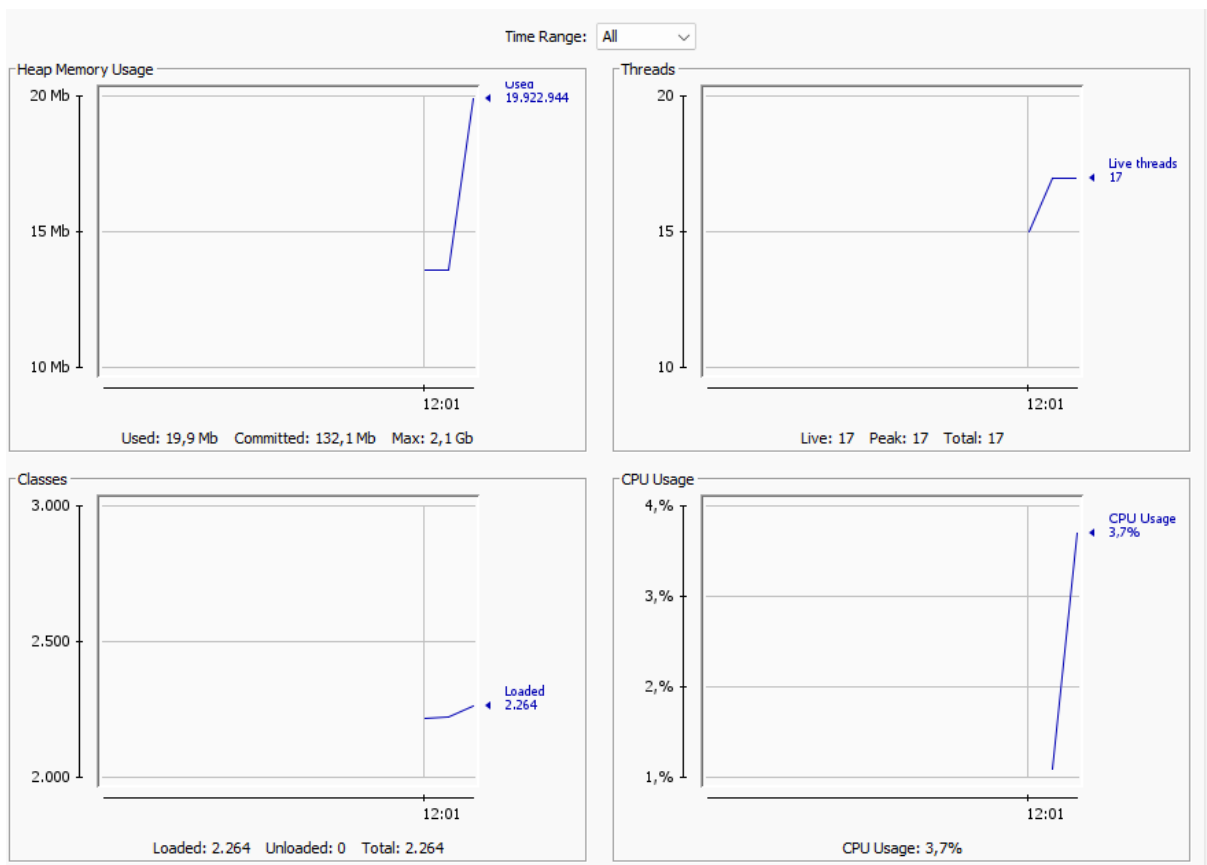
Inserir 10000 AVL



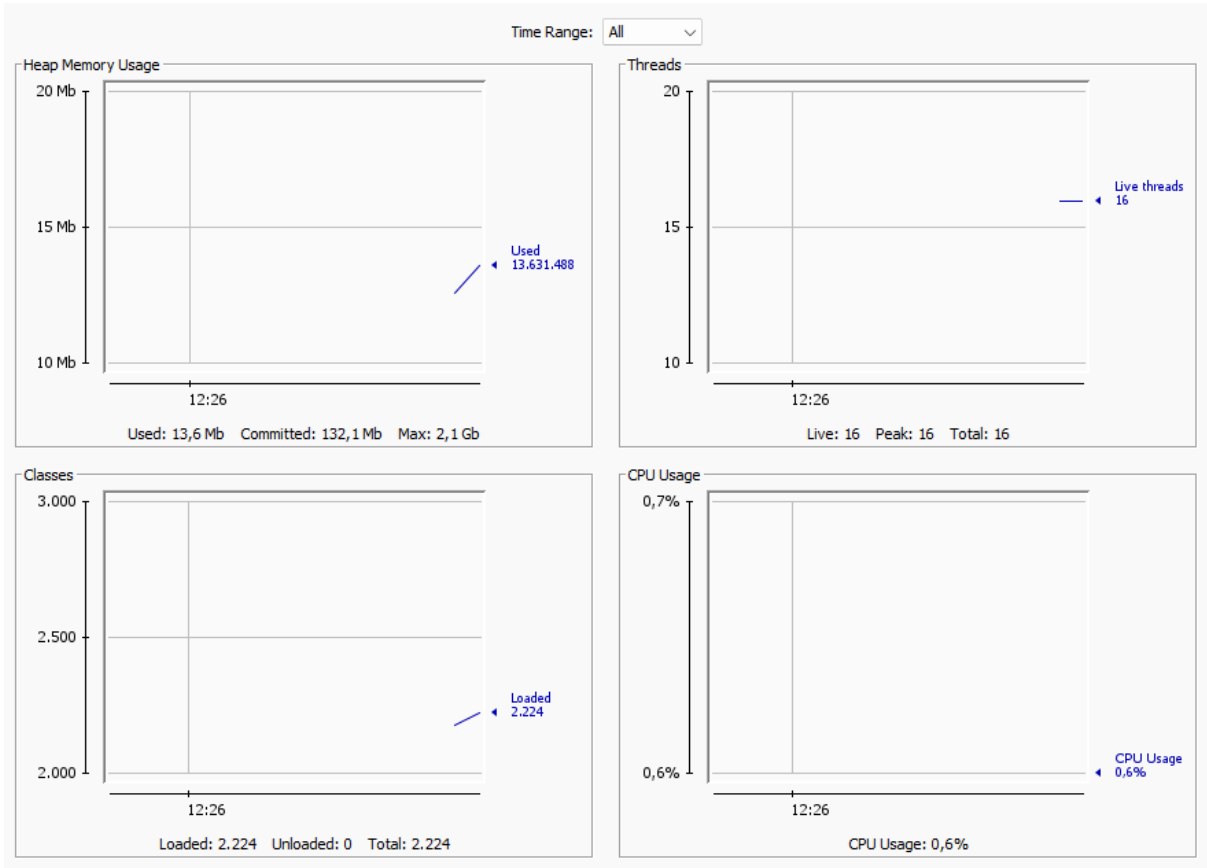
Inserir 20000 AVL



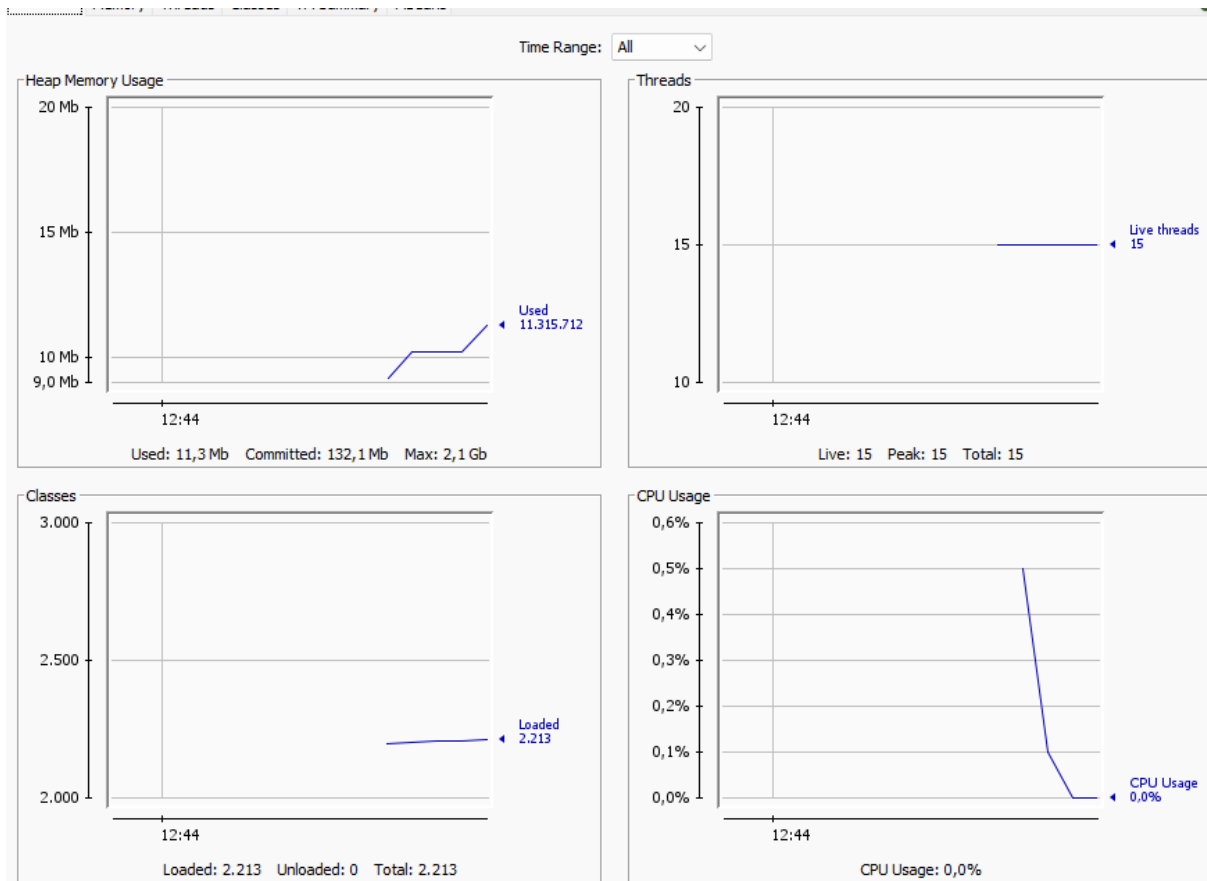
Inserir 1000000 AVL



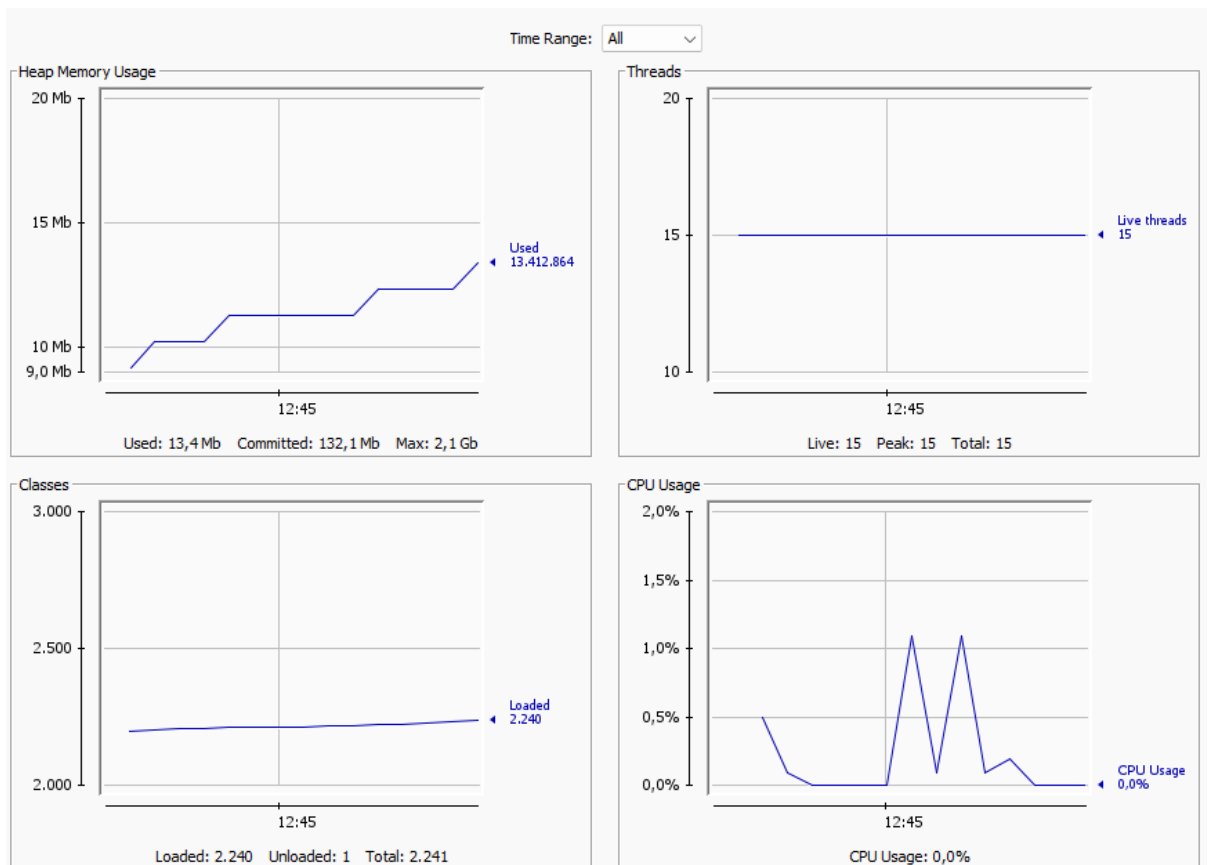
Buscar elemento em árvore de busca (que parece lista encadeada)



Buscar elemento em AVL



Busca elemento árvore de busca

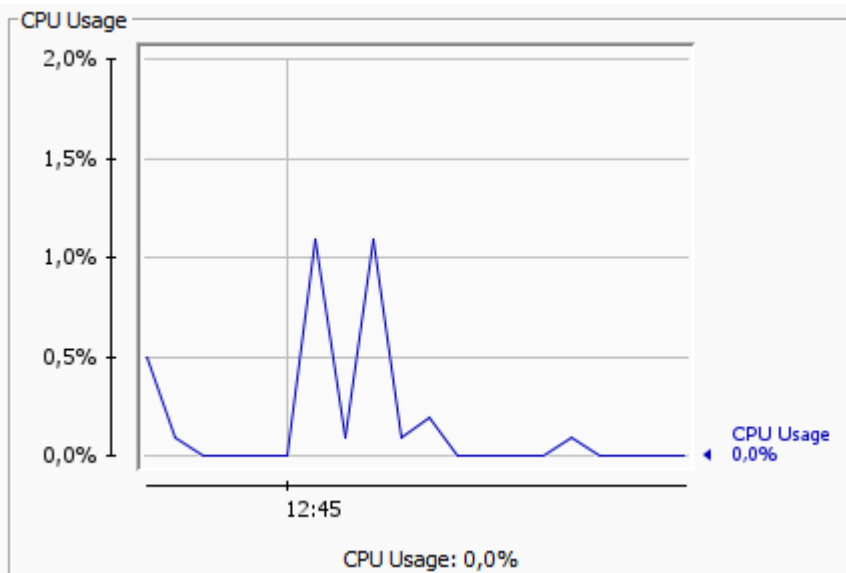


Conclusão da busca

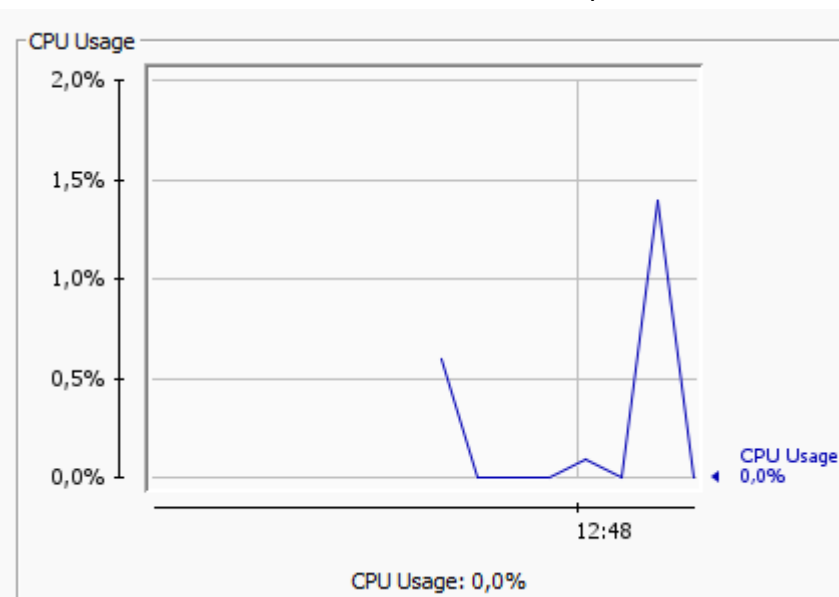
20000 Elementos

Para busca na AVL foi usado 0.5% da CPU

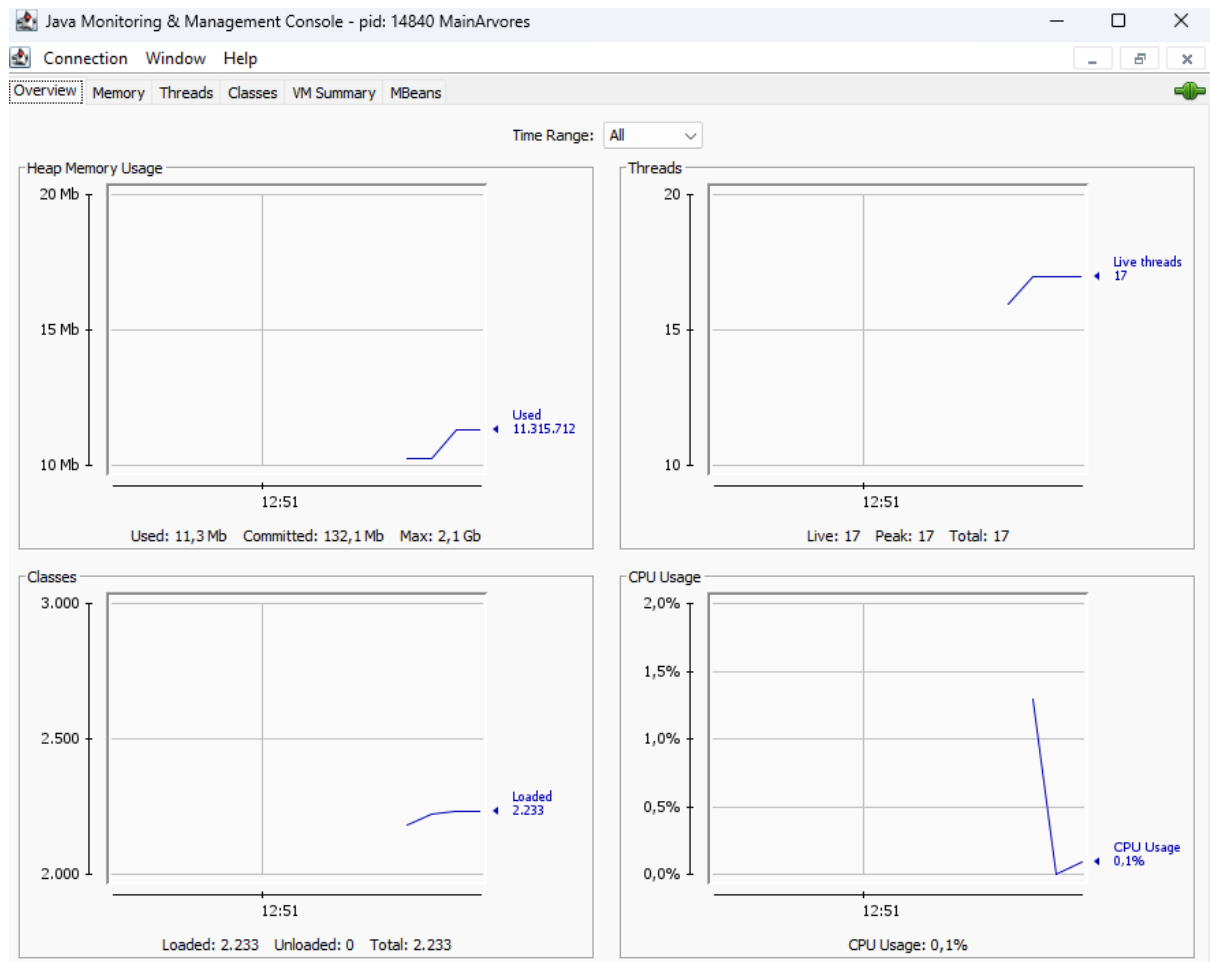
Para busca na árvore de busca foi usado 1% da CPU



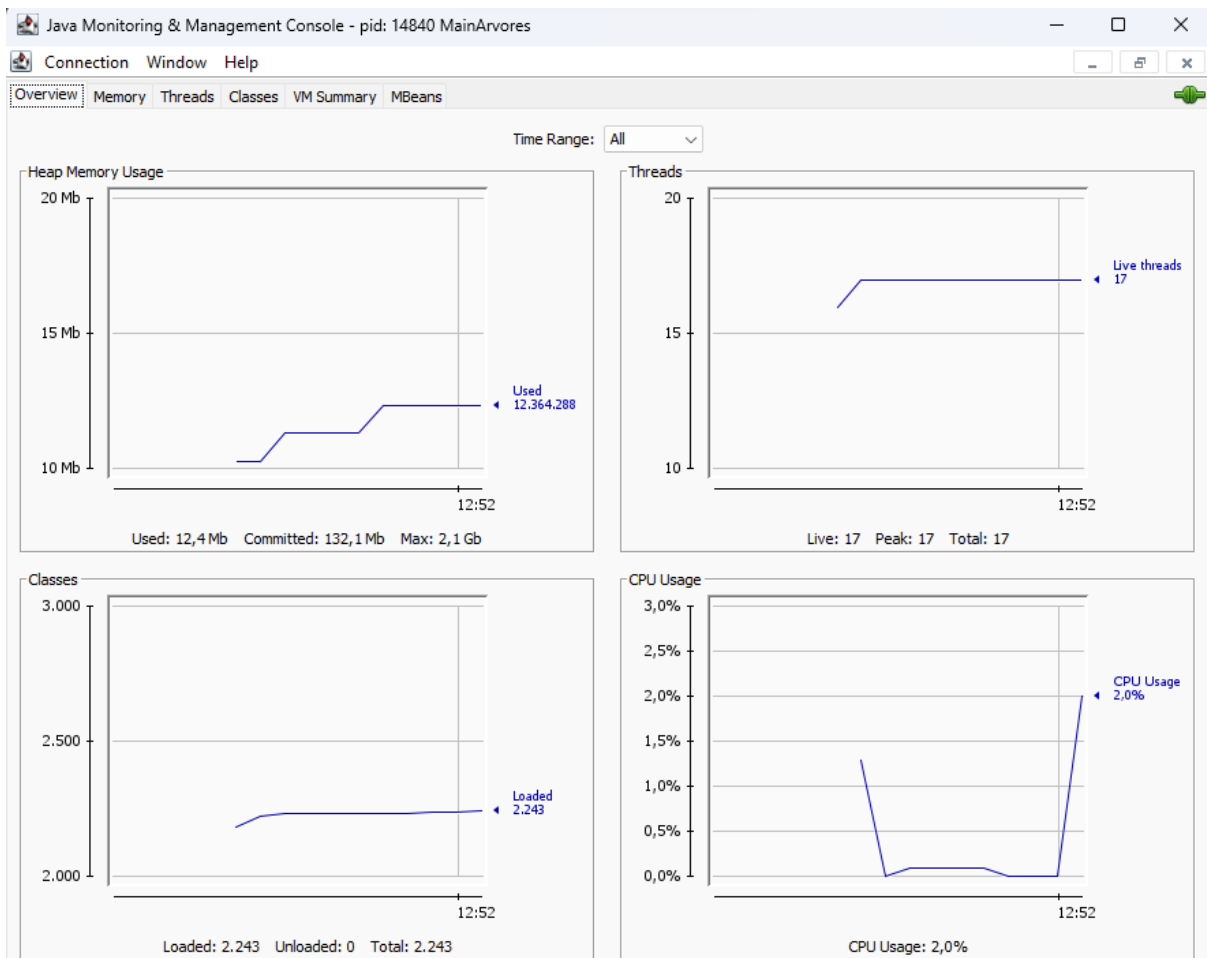
Para 1000 elementos o resultado também foi parecido



Remover na AVL (20000 Elementos)



Remover Busca (20000 Elementos)



Conclusão da remoção:

Como já era esperado, remover na árvore de busca gastou mais recursos computacionais do que a árvore AVL

Código - Busca

INSERIR

Função iterativa para inserir um nó. Se a raiz for nula, define a raiz como o novo nó. Senão, usa um nó auxiliar para percorrer a árvore e encontrar o local adequado para o novo nó, realizando comparação.

BUSCA

Função que retorna o nó da árvore com base em um valor. Para isso verifica se é a raiz, se não for, itera sobre os nós para encontrar o nó correspondente na árvore

BUSCA PAI

Retorna o pai de um nó

REMOVER

Remove um nó da árvore. Para isso, verifica se é raiz, se é nó folha, se tem apenas 1 filho ou se tem 2 filhos. Para cada cenário, ele exclui o nó e adapta a árvore se necessário (caso dos 2 filhos).

Código - AVL

INSERIR

Função iterativa para inserir um nó. Se a raiz for nula, define a raiz como o novo nó. Senão, usa um nó auxiliar para percorrer a árvore e encontrar o local adequado para o novo nó, realizando comparação. Após inserir, percorre a árvore para verificar se os nós estão balanceados.

BUSCA

Função que retorna o nó da árvore com base em um valor. Para isso verifica se é a raiz, se não for, itera sobre os nós para encontrar o nó correspondente na árvore

BUSCA PAI

Retorna o pai de um nó

REMOVER

Remove um nó da árvore. Para isso, verifica se é raiz, se é nó folha, se tem apenas 1 filho ou se tem 2 filhos. Para cada cenário, ele exclui o nó e adapta a árvore se necessário (caso dos 2 filhos). Depois de remover, percorre a árvore para verificar se está balanceada

CALCULAR FB

Retorna o fator de balanceamento de um nó com base em sua altura (alt. sub-árvore esq - alt. sub-árvore dir)

ALTURA

Retorna a altura de um nó

BALANCEAR

Realiza as rotações na árvore em caso de desbalanceamento. Se estiver desequilibrado pra esquerda, e se sim, se está desequilibrado por causa de muitos nós à esquerda. Para desequilíbrio à direita, são realizadas verificações similares.

Conclusão

As árvores AVL, que são balanceadas, vão ser melhor aproveitadas em cenários que exigem manipulação de uma grande massa de dados. Isso ocorre porque a árvore vai estar balanceada, ou seja, em sua estrutura, não terão sub-árvores degeneradas (que são praticamente listas encadeadas). Com isso, será necessário um número menor de iterações, o que resulta em um consumo menor de recursos computacionais.

Por outro lado, as árvores AVL podem ser desnecessárias em cenários que se trabalha com uma quantidade de dados menor, por conta de que os recursos utilizados por ela e a árvore de busca, pelo mesmo cenário, não terão diferenças significativas.

Ou seja, para definir qual vai ser melhor, depende do contexto. Se você tiver uma grande quantidade de informações, use a árvore AVL. Se você for trabalhar com quantidades menores de dados, em um ambiente acadêmico, por exemplo, pode ser mais interessante usar a árvore de busca, já que os recursos expendidos serão os mesmos e é mais fácil de implementar.

Síntese dos resultados obtidos

Operação INSERIR					
Árvore de busca			Árvore AVL		
número de elementos	CPU usage	Heap Memory Usage	número de elementos	Heap Memory Usage	CPU usage
100	1.7	12.6 mb	100	12.6 mb	0.5
500	1.9	12.6 mb	500	12.6 mb	1.1
1000	1.3	12.6 mb	1000	12.6 mb	1.6
10000	1.6	13.6 mb	10000	12.6 mb	1.1
20000	0.8	13.6 mb	20000	12.6 mb	1.0
100000	1.1	13.6	100000	13.6 mb	0.6
1000000	9.8	20.9 mb	1000000	18 mb	1.9
10000000	13.8	400 mb	10000000	59 mb	7.2

Operação BUSCA			
Árvore de busca		Árvore AVL	
número de elementos	CPU usage	Número de elementos	CPU usage
20000	1.0	20000	0.5
100000	1.3	100000	1.1
1000	1.1	1000	0.5
500	0.9	500	0.6
100	1.5	100	1.0
1000	0.5	1000	1.5

Operação REMOVER			
Árvore de busca		Árvore AVL	
número de elementos	CPU usage	Número de elementos	CPU usage
100000	2.1	100000	1.3
20000	2.0	20000	1.3
10000	2.1	10000	1.1
1000	0.7	1000	1.1
500	0.9	500	1.2
100	0.2	100	0.9

