Encontrar menores

Encontrar ***"menores"*** nada mais é que um *algoritmo de comparação*, descobrir o "menor" de uma coleção é comparar todos os elementos em busca do que corresponde o requisito.

Em computação a gente está falando de **guardar o atual na memória (variável)** e **comparar com o próximo da coleção**, essa operação pode ser feita de algumas formas.

Da pra fazer isso usando as **posições da coleção**, onde **a cada iteração** o que **será comparado é o valor, mas o que será armazenado é a posição que tem o menor valor** *(a cada iteração o que é trocado é o valor da posição caso o critério seja atendido):*

```
int posicaoMaisBarata = 0;

for(int posicaoAtual = 0; posicaoAtual < carros.size() ; posicaoAtual ++){
    if(carros.get(posicaoAtual).getPreco().compareTo(carros.get(posicaoMaisBarata).getPreco()) < 0){
        posicaoMaisBarata = posicaoAtual;
    }
}
```

Outra forma seria usando um **enhanced for (um for mais reduzido)**, dessa maneira o código fica muito mais enxuto e legível, mas a única maneira que encontrei pra fazer funcionar é **usando uma variável auxiliar** que estabelece o menor preço com o valor mais alto possível que eu quis usar.

A partir daí cada iteração **vai perguntar se o preço do carro atual é menor do que o da variável auxiliar**, se for ele substitui pelo carro atual e assim sucessivamente *(o que tá sendo comparado aqui é estritamente o preço)*:

```
BigDecimal menorPreco = new BigDecimal( val: "100000000");

for (Carro carro: carros) {
    if(carro.getPreco().compareTo(menorPreco) < 0){
        menorPreco = carro.getPreco();
    }
}
```

E a melhor forma de todas *(na minha opinião)* é usando streams, onde eu **simplesmente pergunto o valor mínimo** usando os **preços dos carros como critério de comparação** e **pego o carro com menor valor** (*aqui eu estou comparando os preços, mas pegando no final o objeto completo de carro, poderia pegar só o preço se fizesse um map):*

```
Carro carroMenorPreco = carros.stream().min(Comparator.comparing(Carro::getPreco)).get();
```

```
Usando variável auxiliar : 16.000
Usando streams (1 linha) : Carro{modelo='Brasilia', preco=16.000}
Usando posições do array: posição do carro 2 valor do carro: 16.000
```