Introdução :

Bubble Sort : É o mais simples que tem, percorre todo o vetor trocando o menor pelo maior. Mais lento.

Selection Sort: Verifica todos os números, seleciona o menor e coloca em primeiro. Melhor para listas/vetores menores.

Inserion Sort : Seleciona o menor número e o coloca antes do número que o precede (ex. 5 antes de 6). Não é muito eficaz em listas muito grandes, pois aumenta a quantidade de trocas que se faz, colocando todos os números atrás do que o precede até chegar na posição correta.

Quick Sort : Seleciona um elemento denominado pivô e organiza a lista que forma que os números anteriores a ele são menores e os posteriores são maiores, repete até ordenar. Um dos métodos mais rápidos e eficaz, pois ordena os 2 lados da lista, tendo 1 número (pivô) como base.

Merge Sort : Divide o vetor em subvetores com no máximo 2 números cada e organiza os subvetores, colocando o menor antes do maior, ao organizar os subvetores, junta eles e organiza de novo até ficar ordenado. É o mais rápido, pois afeta vários números por vez.

Implementação :

A principal implementação que encontrei foi usar o *for* dentro do *for* com um *if* dentro do segundo *for* para controlar quando ele é acionado e quando não é. No *merge sort* em especial, foi usado *while* e *else* para controlar os subvetores e, trocar suas posições usando o *temp* e o método de troca de vetores.

Análise de Complexidade :

Método	Complexidade
Inserção	$O(n^2)$
Seleção	$O(n^2)$
Bolha	$O(n^2)$
Shellsort	$O(n\ lg(n)^2)$
Quicksort	$O(n\ lg(n))$
Heapsort	$O(n\ lg(n))$
Radixsort	$O(kn)$

- O método que levou menos tempo real para executar recebeu o valor 1 e os outros receberam valores relativos
- Elementos em ordem aleatória:

	5.00	5.000	10.000	30.000
Inserção	11,3	87	161	–
Seleção	16,2	124	228	–
Shellsort	1,2	1,6	1,7	2
Quicksort	1	1	1	1
Heapsort	1,5	1,6	1,6	1,6

Elementos em ordem crescente

	500	5.000	10.000	30.000
Inserção	1	1	1	1
Seleção	128	1.524	3.066	–
Shellsort	3,9	6,8	7,3	8,1
Quicksort	4,1	6,3	6,8	7,1
Heapsort	12,2	20,8	22,4	24,6

Elementos em ordem decrescente

	500	5.000	10.000	30.000
Inserção	40,3	305	575	–
Seleção	29,3	221	417	–
Shellsort	1,5	1,5	1,6	1,6
Quicksort	1	1	1	1
Heapsort	2,5	2,7	2,7	2,9

Conclusão : A principal dificuldade que eu tive foi saber como eu implementaria algo que eu nunca vi na vida, e como eu aprenderia por mim mesmo aquilo baseado em exemplos e poucas explicações da internet. O resultado que encontrei foi alguns tiveram bastante dificuldade de executar tantos vetores, como o bubble sorte por exemplo.