Unidade I:

Introdução - Algoritmos de Seleção

Analise do número de movimentações:

```
for (int i = 0; i < (n - 1); i++) {
    int menor = i;
    for (int j = (i + 1); j < n; j++){
        if (array[menor] > array[j]){
            menor = j;
        }
    }
    swap(menor, i);
}
```

a) Quantas movimentações (entre elementos do array) são realizadas?
R. Swap realiza 3 movimentações, ou seja, f(n) = 3(n-1)

Exercício Resolvido (1):

b) Faça com que nosso código conte o número de movimentações?
R.
Obs. declarar **int mov = 0** no começo do método

```
...
7º linha}
        mov+=3;
        swap(menor, i);
    }
    System.out.println(mov);
```

c) Quantas comparações (entre elementos do array) são realizadas?
R. São realizadas n-1(i-1) = (n - i - 1)

Exercício Resolvido (2):

Na pasta da “Unidade 4/c/”, digite a sequência de comandos abaixo e explique a saída na tela.

1) make all ; ls
2) make clean ; ls
3) make ; ls
4) make clean ; ls
5) make exec ; ls
6) make limpa ; ls
7) make bolha.o ; ls

Resposta:

a)	gcc -o principal.o principal.c -c -W -Wall -pedantic gcc -o geracao.o geracao.c -c -W -Wall -pedantic gcc -o bolha.o bolha.c -c -W -Wall -pedantic gcc -o countingsort.o countingsort.c -c -W -Wall -pedantic gcc -o heapsort.o heapsort.c -c -W -Wall -pedantic gcc -o insercao.o insercao.c -c -W -Wall -pedantic gcc -o mergesort.o mergesort.c -c -W -Wall -pedantic gcc -o quicksort.o quicksort.c -c -W -Wall -pedantic gcc -o selecao.o selecao.c -c -W -Wall -pedantic gcc -o shellsort.o shellsort.c -c -W -Wall -pedantic gcc -o exec principal.o geracao.o bolha.o countingsort.o heapsort.o insercao.o mergesort.o quicksort.o selecao.o shellsort.o bolha.c bolha.o countingsort.h exec geracao.h heapsort.c heapsort.o insercao.h makefile mergesort.h principal.c quicksort.c quicksort.o selecao.h shellsort.c shellsort.o bolha.h countingsort.c countingsort.o geracao.c geracao.o heapsort.h insercao.c insercao.o mergesort.c mergesort.o principal.o quicksort.h selecao.c selecao.o shellsort.h Compila todos arquivos
b)	rm -rf *.o *~ exec bolha.c bolha.h countingsort.c countingsort.h geracao.c geracao.h heapsort.c heapsort.h insercao.c insercao.h makefile mergesort.c mergesort.h principal.c

	quicksort.c quicksort.h selecao.c selecao.h shellsort.c shellsort.h Remove todos os arquivos compilados (que possuem extensão .o)
c)	gcc -o principal.o principal.c -c -W -Wall -pedantic gcc -o geracao.o geracao.c -c -W -Wall -pedantic gcc -o bolha.o bolha.c -c -W -Wall -pedantic gcc -o countingsort.o countingsort.c -c -W -Wall -pedantic gcc -o heapsort.o heapsort.c -c -W -Wall -pedantic gcc -o insercao.o insercao.c -c -W -Wall -pedantic gcc -o mergesort.o mergesort.c -c -W -Wall -pedantic gcc -o quicksort.o quicksort.c -c -W -Wall -pedantic gcc -o selecao.o selecao.c -c -W -Wall -pedantic gcc -o shellsort.o shellsort.c -c -W -Wall -pedantic gcc -o exec principal.o geracao.o bolha.o countingsort.o heapsort.o insercao.o mergesort.o quicksort.o selecao.o shellsort.o bolha.c bolha.o countingsort.h exec geracao.h heapsort.c heapsort.o insercao.h makefile mergesort.h principal.c quicksort.c quicksort.o selecao.h shellsort.c shellsort.o bolha.h countingsort.c countingsort.o geracao.c geracao.o heapsort.h insercao.c insercao.o mergesort.c mergesort.o principal.o quicksort.h selecao.c selecao.o shellsort.h Compila todos arquivos
d)	rm -rf *.o *~exec bolha.c bolha.h countingsort.c countingsort.h geracao.c geracao.h heapsort.c heapsort.h insercao.c insercao.h makefile mergesort.c mergesort.h principal.c quicksort.c quicksort.h selecao.c selecao.h shellsort.c shellsort.h Remove todos os arquivos compilados (que possuem extensão .o).
e)	gcc -o principal.o principal.c -c -W -Wall -pedantic gcc -o geracao.o geracao.c -c -W -Wall -pedantic gcc -o bolha.o bolha.c -c -W -Wall -pedantic gcc -o countingsort.o countingsort.c -c -W -Wall -pedantic gcc -o heapsort.o heapsort.c -c -W -Wall -pedantic gcc -o insercao.o insercao.c -c -W -Wall -pedantic gcc -o mergesort.o mergesort.c -c -W -Wall -pedantic gcc -o quicksort.o quicksort.c -c -W -Wall -pedantic gcc -o selecao.o selecao.c -c -W -Wall -pedantic gcc -o shellsort.o shellsort.c -c -W -Wall -pedantic

	gcc -o exec principal.o geracao.o bolha.o countingsort.o heapsort.o insercao.o mergesort.o quicksort.o selecao.o shellsort.o bolha.c bolha.o countingsort.h exec geracao.h heapsort.c heapsort.o insercao.h makefile mergesort.h principal.c quicksort.c quicksort.o selecao.h shellsort.c shellsort.o bolha.h countingsort.c countingsort.o geracao.c geracao.o heapsort.h insercao.c insercao.o mergesort.c mergesort.o principal.o quicksort.h selecao.c selecao.o shellsort.h Compila todos arquivos
f)	rm -rf *.o bolha.c bolha.h countingsort.c countingsort.h geracao.c geracao.h heapsort.c heapsort.h insercao.c insercao.h makefile mergesort.c mergesort.h principal.c quicksort.c quicksort.h selecao.c selecao.h shellsort.c shellsort.h Remove todos os arquivos compilados (que possuem extensão .o) deixando os arquivos executáveis.
g)	gcc -o bolha.o bolha.c -c -W -Wall -pedantic bolha.c bolha.h bolha.o countingsort.c countingsort.h exec geracao.c geracao.h heapsort.c heapsort.h insercao.c insercao.h makefile mergesort.c mergesort.h principal.c quicksort.c quicksort.h selecao.c selecao.h shellsort.c shellsort.h Compila somente bolha.o.

Exercício (1):

Mostre todas as comparações e movimentações do algoritmo anterior para o array abaixo:

12	4	8	2	14	17	6	18	10	16	15	5	13	9	1	11	7	3

Resposta:

12	4	8	2	14	17	6	18	10	16	15	5	13	9	1	11	7	3
1	4	8	2	14	17	6	18	10	16	15	5	13	9	12	11	7	3
1	2	8	4	14	17	6	18	10	16	15	5	13	9	12	11	7	3
1	2	3	4	14	17	6	18	10	16	15	5	13	9	12	11	7	8
1	2	3	4	14	17	6	18	10	16	15	5	13	9	12	11	7	8
1	2	3	4	5	17	6	18	10	16	15	14	13	9	12	11	7	8
1	2	3	4	5	6	17	18	10	16	15	14	13	9	12	11	7	8
1	2	3	4	5	6	7	18	10	16	15	14	13	9	12	11	17	8
1	2	3	4	5	6	7	8	10	16	15	14	13	9	12	11	17	18
1	2	3	4	5	6	7	8	9	16	15	14	13	10	12	11	17	18
1	2	3	4	5	6	7	8	9	10	15	14	13	16	12	11	17	18
1	2	3	4	5	6	7	8	9	10	11	14	13	16	12	15	17	18
1	2	3	4	5	6	7	8	9	10	11	12	13	16	14	15	17	18
1	2	3	4	5	6	7	8	9	10	11	12	13	14	16	15	17	18
1	2	3	4	5	6	7	8	9	10	11	12	13	14	16	15	17	18
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Verde = Grupo ordenado	Azul = ação atual de movimentação

Exercício (2):

Execute a versão abaixo do Seleção para arrays gerados aleatoriamente. Em seguida, discuta sobre os números de comparações inseridas e movimentações evitadas pela nova versão do algoritmo:

```
for (int i = 0; i < (n - 1); i++) {
    int menor = i;
    for (int j = (i + 1); j < n; j++){
        if (array[menor] > array[j]){
            menor = j;
        }
    }
    if (menor != i){
        swap(menor, i);
    }
}
```

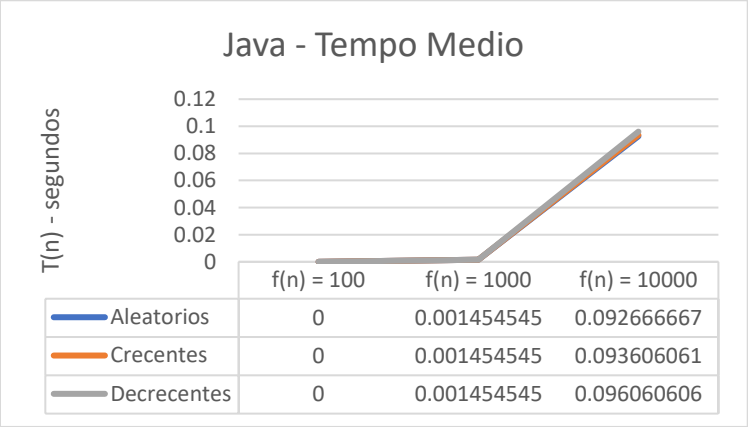
Resposta:

Caso o **menor** for igual a **i**, não haverá nenhuma movimentação, porque **menor** já está na posição **i**, assim fazemos com que não haja movimentações desnecessárias.

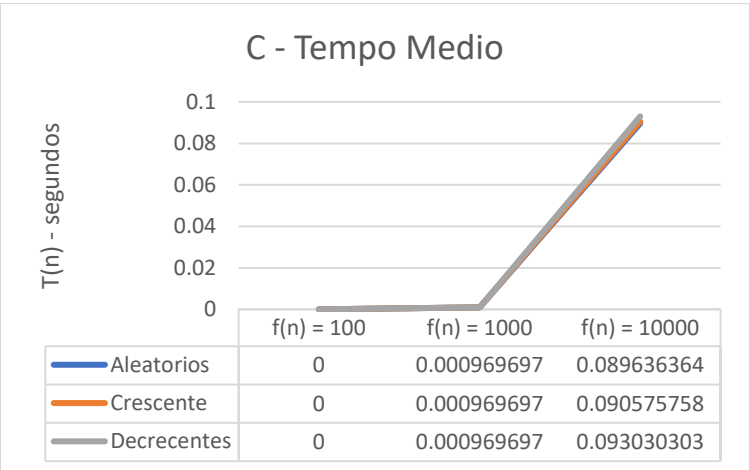
Exercício (3):

Contabilize os números de comparações e movimentações entre elementos do array; calcule os valores teóricos para as duas métricas; e contabilize o tempo de execução. Em seguida, para os códigos em Java e C, gere arrays aleatórios (seed 0) com tamanhos 100, 1000 e 10000. Para cada instância (variação de linguagem e tamanho de vetor), faça 33 execuções. Faça um gráfico para os valores médios de cada métrica avaliada (comparações, movimentações e tempo de execução) variando o tamanho do array. Nos gráficos de comparações e movimentações, mostre também os resultados teóricos. Cada gráfico terá uma curva para cada linguagem. Interprete os resultados obtidos. Repita o processo para arrays gerados de de forma crescente e decrescente.

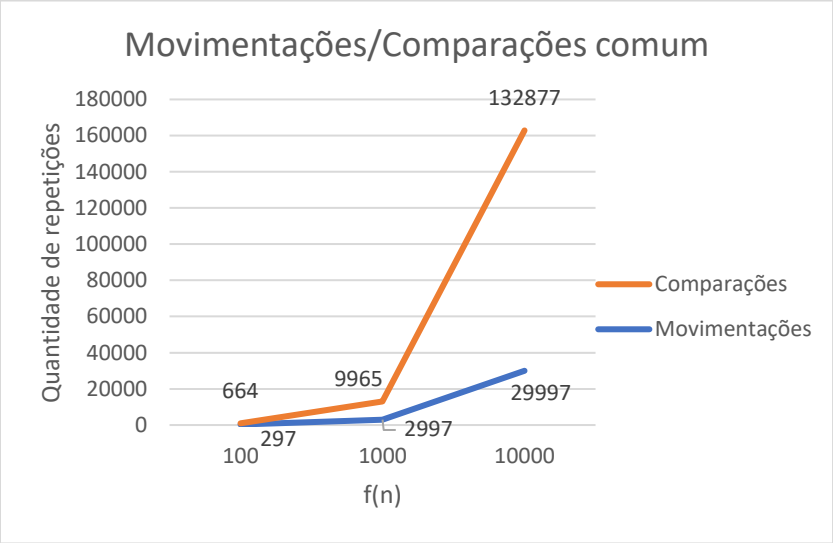
Resposta:



	f(n) = 100	f(n) = 1000	f(n) = 10000
Aleatorios	0	0.001454545	0.092666667
Crecentes	0	0.001454545	0.093606061
Decrecentes	0	0.001454545	0.096060606



	f(n) = 100	f(n) = 1000	f(n) = 10000
Aleatorios	0	0.000969697	0.089636364
Crescente	0	0.000969697	0.090575758
Decrecentes	0	0.000969697	0.093030303


Movimentações/Comparações comum
Quantidade de repetições
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
132877
9965
664
297
2997
29997
100
1000
10000
f(n)
Comparações
Movimentações