Aula Prática 9

Resumo:
- Complexidade algorítmica;
- Algoritmos de ordenação;
- Recursividade.

Exercício 9.1

Implemente o algoritmo de ordenação por fusão (*Merge Sort*) para números inteiros. Utilize o programa `P91.java`[1] para testar a função. Podemos utilizar este programa de inúmeras formas. Por exemplo:

```
$ # input as output of command echo (piping):
$ echo "4 3 2 6 1 2 3 5 7 8 9 5" | java -ea P91

$ # input as output of command cat (piping):
$ cat numbers.txt | java -ea P91

$ # redirecting input from file (assuming file numbers.txt exists):
$ java -ea P91 < numbers.txt

$ # no piping nor redirecting (Ctrl-d to terminate the input):
$ java -ea P91
4 5 2 3 2
2 3
<Ctrl-d>
```

Exercício 9.2

Implemente o algoritmo de ordenação por inserção para números inteiros. Utilize o programa `P92.java` para testar a função.

Exercício 9.3

O programa `P93.java` mede os tempos necessários para ordenar arrays de N inteiros,

[1] Disponível dentro do arquivo `aula09.zip` existente no `moodle`.

usando dois algoritmos que já conhece: a ordenação sequencial e a ordenação "bolha". Experimente-o e analise-o para perceber o seu funcionamento.

a. O programa inclui dois campos `static` para contabilizar operações elementares:

   `comparisonCount` para contar comparações entre elementos do array;

   `assignCount` para contar atribuições valor a elementos.

   Acrescente instruções para incrementar esses contadores nos locais apropriados dos algoritmos. Teste o programa com $N = 100$ e $N = 1000$ e observe as contagems de operações nos vários casos.

b. Acrescente as funções de ordenação por inserção e por fusão. Inclua instruções para as contagens de operações. Volte a testar para comparar os resultados.

c. Teste o programa para vários valores de N e escreva uma tabela com os resultados das diversas medições para todos os algoritmos de ordenação, e para os três casos de teste (arrays aleatórios, ordenados crescentemente e ordenados decrescentemente).

Exercício 9.4

A função `extraiPalavras` (programa `P94.java`) extrai para um *array* todas as palavras existentes num ficheiro de texto. Construa um programa que crie uma lista ordenada de todas as palavras existentes em ficheiros de texto (argumentos do programa) eliminando todas as repetições de palavras. O programa deve escrever a lista ordenada de palavras encontradas em todos os ficheiros e indicar o seu número.

Exercício 9.5

Altere o programa anterior por forma a que a informação por ele gerada passe a ser persistente. Assim, no fim da execução do programa a lista de palavras (sem repetição) por ele gerada deverá ser registada num ficheiro de texto (`words.txt`), e a (eventual) lista de palavras existente nesse ficheiro deverá ser lida pelo programa no início da sua execução.

Exercício 9.6

O programa `ListSort` lê um ou mais ficheiros de texto e ordena as suas linhas. Para isso, armazena as linhas numa lista ligada e depois ordena-as usando uma versão do algoritmo QuickSort. Preencha as lacunas na função de ordenação para completar o programa.