UFMG - ICEx - DCC

Algoritmos e Estruturas de Dados II

Aula Prática 1 - Complexidade

Descrição

Nesta aula prática vamos exercitar a análise de complexidade de algoritmos.

Objetivo

O nosso objetivo é implementar e analisar a complexidade de três algoritmos simples. Dois deles são usados para determinar os valores máximos e mínimos de uma sequência de números aleatórios, enquanto o outro serve para ordenar essa sequência.

Especificação

Nesta aula prática você deve implementar em C:

1. um algoritmo para determinar os valores máximo e mínimo da sequência que faça $2(n-1)$ comparações;
2. um algoritmo para determinar os valores máximo e mínimo da sequência que faça $\frac{3n}{2}$ comparações;
3. o algoritmo de ordenação por Inserção (*Insertion Sort*).

Entrada

Seu programa deve ler da entrada padrão um número inteiro positivo $1 \leq n \leq 100$ que representa o tamanho da sequência, seguido pelos n valores que a compõe (separados por espaço em branco).

Saída

A primeira linha da saída deve conter os valores máximo e mínimo da sequência separados por um espaço obtidos utilizando o primeiro algoritmo implementado.
A segunda linha de saída deve conter os valores máximo e mínimo da sequência separados por espaço obtidos utilizando o segundo algoritmo implementado.
A terceira linha de saída deve conter os valores da sequência em ordem crescente separados por espaço (inclusive o último valor deve ter um espaço antes da quebra de linha).

Exemplo

Execução:

```
$ ./main
```

Entrada:

```
8 3796 4822 -2165 2703 -2233 -2891 -4189 -2050
```

Saída:

```
-4189 4822
-4189 4822
-4189 -2891 -2233 -2165 -2050 2703 3796 4822
```

Material

São fornecidos:

1. **main.c**: arquivo base para a implementação do seu programa. Os três algoritmos requeridos devem ser implementados neste arquivo;

2. **Makefile**: o Makefile é um arquivo que auxilia a compilação do seu programa (em ambientes Linux). Basta colocá-lo na mesma pasta do arquivo main.c e usar o comando `make` que o executável `main` será gerado:

   ```
   $ make
   gcc -o main main.c -Wall -Werror -ansi -pedantic
   ```

3. **testes**: diretório contendo alguns casos de teste. Os arquivos test#.in são arquivos de entrada, enquanto que os test#.res são as saídas correspondentes a cada arquivo de entrada. Em um ambiente Linux você pode utilizar esses arquivos para verificar se seu programa responde da forma esperada pelo software de correção.

   ```
   $ ./main < testes/test4.in
   ```

   executará seu programa para o caso de teste contido em `test4.in`.

   ```
   -4189 4822
   -4189 4822
   -4189 -2891 -2233 -2165 -2050 2703 3796 4822
   ```

   Em um ambiente Linux você pode utilizar o utilitário `diff` para comparar exatamente a saída do seu programa com a saída esperada pelo corretor automático. Por exemplo, caso de sucesso:

   ```
   $ ./main < testes/test4.in | diff testes/test4.res -
   ```

   e caso de erro:

   ```
   $ ./main < testes/test4.in | diff testes/test4.res -
   1c1
   < -4189 4822
   ---
   > 4822 -4189
   ```

   Dica: Quando a saída de seu programa parece estar igual ao resultado do caso de teste e mesmo assim o `diff` acusa diferenças, tente direcionar a saída do seu programa para um arquivo e utilizar o utilitário `meld` para apontar graficamente as diferenças (`meld arq1 arq2`).

Entrega

O trabalho deve ser feito em dupla. Cada dupla deverá entregar um arquivo zip nomeado com suas matrículas e contendo os arquivos fonte para execução do programa. No caso desta aula prática, apenas o arquivo **main.c**. Por exemplo, os alunos cujas matrícula são 201700000001 e 201700000002, devem enviar o arquivo *201700000001-201700000002.zip*, contendo seu arquivo solução **main.c**.

Além disso, a dupla deve utilizar o endereço indicado na descrição da aula prática no Moodle para acessar e preencher um formulário com questões sobre o comportamento assintótico dos algoritmos implementados.

Outras Instruções

A data para conclusão da aula prática está na descrição da aula no Moodle. Eventuais dúvidas deverão ser postados no fórum do Moodle e serão respondidas por lá.

Os exercícios das aulas práticas serão corrigidos de forma automática, tenha certeza de que a saída produzida por seu programa seja a mesma esperada pelo software de correção. Você pode se basear nos casos de teste enviados para comparar sua resposta com a resposta esperada. Além dos casos de testes enviados, serão executados outros testes.

É altamente aconselhável que os alunos troquem experiências e conversem sobre os exercícios práticos, contudo as atividades, a menos que informado explicitamente, são em dupla e eventuais casos de plágio ou cópias de trabalhos de outras duplas serão punidos. Para tal, o software de correção utilizado é equipado com algoritmos de detecção de plágio, portanto, implementem suas próprias soluções.

Bom Trabalho!