Projeto 5 - Ordenação

ATP II

1 Definição

Números inteiros (sem usar long int) são representados corretamente até o valor de 2.147.483.647, que é o limite de representação atual de inteiros. Considerando isso, se formos tratar de inteiros maiores que esse valor temos três opções. Uma é usar o tipo *long long int*, que vai até 9.223.372.036.854.775.807 (e novamente temos problema para valores maiores que 10^{19}). Outra opção é transformar o inteiro em um tipo real, como float/double, porém isso nem sempre é desejável. A outra forma é criando um tipo de dados que concatene dois inteiros, de modo a formar um inteiro muito grande.

Isso é similar ao que fazemos separando um número em unidades, dezenas, centenas, milhares, etc. Ou seja, podemos pensar que um número inteiro grande é a combinação de dois inteiros, um representando a "parte baixa" (no caso sem estourar o valor de um bilhão, 1.000.000.000) e o outro representando a "parte alta". Assim, o valor 100 bilhões seria representado por um inteiro 100 e por um inteiro 0. Já o valor limite para inteiros indicado acima seria representado por 2 e por 147483647.

Para fazer a representação indicada (esqueça das funções que manipulam esses valores), deve se declarar a seguinte estrutura

```
typedef struct biggo {
  int high; int low; } BigInt;
```

2 O que deve ser feito

Considerando então esse novo tipo de dados, implemente o algoritmo quicksort sort, sem usar a função da biblioteca do C, para ordenar um vetor de 200 mil posições BigInt. Lembre-se nesse caso que a ordenação deve cuidar para que posições de mesmo valor para *high* sejam ordenadas pelo seu valor de *low*.

O seu programa lerá os dados de um arquivo chamado bigint.dat e deve gerar como resposta um arquivo chamado quick.dat.

Para efeito de comparação, use também o algoritmo insertion sort (devem ser executados separadamente, em programas distintos).

A entrega do programa deve incluir o código fonte e um pequeno relato dos resultados obtidos (1 ou 2 páginas).

3 Entrada de dados

O programa deve ler os dados de um arquivo chamado bigint.dat, que contém 200 mil valores do tipo BigInt, sendo um valor (a parte high e a parte low) por linha.

4 Saída de dados

Seu programa deve produzir um arquivo (quick.dat), contendo os 200 mil valores BigInt ordenados de forma crescente.

Como indicado anteriormente, deverá também ser implementado o método de ordenação insertion sort.

Você deverá executar o programa pelo menos 5 vezes para cada método de ordenação, medindo o tempo gasto apenas com a ordenação, apresentando um relatório que mostre a diferença de tempo observada.

5 Exemplo

Um possível começo de arquivo de entrada seria:

Entrada
3 400300200
1 737779000
-36 987654321
-36 200000000
3 100300
...

Saída
-36 987654321
-36 200000000
1 737779000
3 100300
3 400300200
...

Observe que a formatação esquisita não muda eventuais operações aritméticas sobre os números BigInt.

6 Entrega

Entregar o código fonte do programa, devidamente comentado, e também o pequeno relatório no classroom.

PRAZO: 26/01, até 23h59 no classroom.