PROVA DE RECUPERAÇÃO DE PRINCÍPIOS DE DESENVOLVIMENTO DE ALGORITMOS
2o. SEMESTRE DE 2006

Instruções:

1. Não destaque as folhas do caderno de soluções.
2. A prova pode ser feita a lápis. Cuidado com a legibilidade.
3. Não é permitido o uso de folhas avulsas para rascunho.
4. Não é necessário apagar rascunhos no caderno de soluções.
5. Asserções imprecisas valem pouco. Justifique suas asserções (dentro do razoável). Seja claro e objetivo.

1. [3 pontos]
   (*i*) Descreva precisamente o algoritmo de ordenação *mergesort* (ordenação por intercalação).
   (*ii*) Quanto tempo demora o *mergesort* para ordenar um vetor de inteiros com n entradas? Justifique sua resposta cuidadosamente.
   (*iii*) Descreva precisamente o algoritmo de ordenação *quicksort*.
   (*iv*) Quanto tempo demora o *quicksort* para ordenar um vetor de inteiros com n entradas? Justifique sua resposta cuidadosamente.
2. [3 pontos] Nesta disciplina, estudamos *tabelas de símbolos*: estruturas de dados para armazenar *itens* que admitem as operações de inserção e busca de itens por *chaves*, entre outras.
   (*i*) Descreva precisamente como implementar uma tabela de símbolos usando *árvores binárias de busca* (ABBs).
   (*ii*) Discuta as características principais de desempenho de tabelas de símbolos implementadas com ABBs (por exemplo, quanto tempo demora tipicamente a inserção de um item em uma tal tabela, supondo que há na tabela n itens?). Justifique sua resposta cuidadosamente, explicitando as hipóteses em sua análise.
   (*iii*) Descreva precisamente como implementar uma tabela de símbolos usando *tabelas de hashing*.
   (*iv*) Discuta as características principais de desempenho de tabelas de símbolos implementadas com tabelas de hashing (por exemplo, quanto tempo demora tipicamente a inserção de um item em uma tal tabela, supondo que há na tabela n itens?). Justifique sua resposta cuidadosamente, explicitando as hipóteses em sua análise.

3. [4 pontos] Descreva o projeto de um programa que recebe uma cadeia de caracteres T como entrada e devolve o trecho repetido mais comprido neste texto. Por exemplo, se o texto de entrada for `aacabcabcbabcbabac`, então a saída de seu programa deve ser `abcbab`.

   Faça uma descrição cuidadosa de seu projeto de programa: descreva a estrutura de dados a ser usada e dê os protótipos das funções principais, com uma descrição de o que estas funções devem fazer e como elas poderiam ser implementadas (não esqueça do `main()`).

   O seu projeto deve supor que a entrada T pode ser grande; por exemplo, T poderia um livro todo. Um programa implementado de acordo com o seu projeto deve ser tal que (*i*) sob hipóteses razoáveis, ele leva tempo basicamente não muito maior que proporcional a $n \log n$, onde n é o número de caracteres em T,

(*ii*) ele gasta uma quantidade de memória proporcional ao número de caracteres em T.

Você deve argumentar por que o programa teria estas propriedades (em particular, você deve explicitar as hipóteses que você usa em sua análise).