MAC 5710 - Estruturas de Dados e sua Manipulação

Primeiro semestre de 2001 – Segunda Prova

Nome do aluno: ____

Assinatura: ____

Instruções:

1. Destaque apenas a última folha deste caderno.
2. Justifique as suas respostas.
3. Preencha o cabeçalho acima.
4. A prova consta de 5 questões. Verifique antes de começar a prova se o seu caderno de questões está completo.
5. A duração da prova é 2 horas.

BOA PROVA!

Questão	Nota
1	
2	
3	
4	
5	
Total	

1. (Valor: 2,0 pontos)

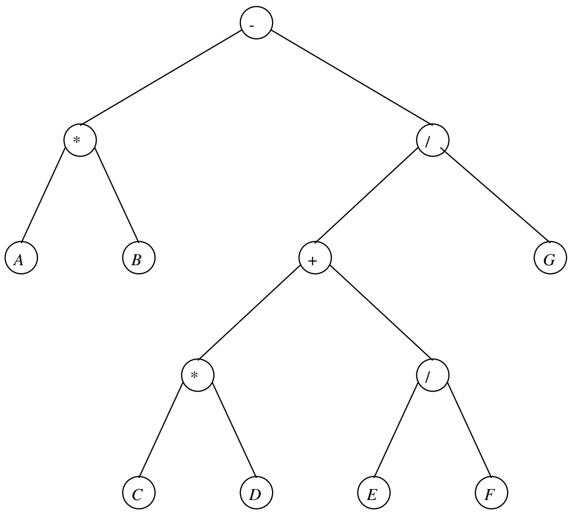
   (a) Em um *heap* de máximo, em que posições pode estar o elemento mínimo?

   (b) Quais das seqüências abaixo são *heaps*?

(i)	23	17	14	6	13	10	1	5	7	12
(ii)	20	7	18	5	6	12	15	2	1	4
(iii)	28	27	26	25	24	12	13	14	15	16

2. (Valor: 3,0 pontos) Suponha que uma expressão aritmética envolvendo os operadores $+$, $-$, $*$ e $/$ esteja armazenada em uma árvore binária da seguinte maneira. Os nós internos da árvore armazenam operadores e as folhas armazenam variáveis. Cada variável é representada por uma letra. Os operandos do operador guardado em um nó interno x são respectivamente as expressão armazenada na subárvore esquerda de x e a expressão armazenada na subárvore direita de x.

   Exemplo: A expressão $A * B - (C * D + E/F)/G$ ficaria



   Suponha as seguintes declarações:

```
struct no {
  char info;
  struct no *esq, *dir;
};

double valor (char ch);
```

   A função `valor` recebe uma letra e devolve o valor da variável representada por aquela letra.

   (a) Escreva uma função `avalia` que recebe como parâmetro um apontador para a raiz de uma árvore que armazena uma expressão e devolve o valor da expressão.

   (b) Escreva uma função `imprime` que recebe como parâmetro um apontador para a raiz de uma árvore que armazena uma expressão e imprime a expressão em notação posfixa.

3. (Valor: 1,5 pontos) Mostre que, em uma árvore rubro-negra, um caminho de um nó x até uma folha abaixo de x tem comprimento no máximo duas vezes o comprimento de qualquer outro caminho de x até uma folha abaixo de x.

4. (Valor: 2,5 pontos) Considere um tipo abstrato de dados que admite as seguintes operações.

   `insere`(x): insere um elemento x;

   `remove`(y): remove um elemento **qualquer**, devolvendo uma cópia dele em y.

   Existem duas possibilidades: o tipo pode permitir duplicatas ou não. Se duplicatas são permitidas, a inserção deve ser feita independente de x já estar presente ou não, e na remoção apenas uma das cópias do elemento escolhido deve ser removida. Se duplicatas não são permitidas, a função `insere` deve verificar se x está presente e inseri-lo apenas se ele não estiver presente.

   (a) Descreva uma estrutura de dados o mais eficiente possível para cada um dos casos.

   Com duplicatas:

   Sem duplicatas:

   (b) Qual o custo de cada uma das duas operações, `insere` e `remove`, na sua estrutura de dados? ($O(n^2)$? $O(n \log n)$? $O(n)$? $O(\log n)$? $O(1)$?)

	`insere`	`remove`
Com duplicatas		
Sem duplicatas		

MAC 5710 - Estruturas de Dados e sua Manipulação

Primeiro semestre de 2001 – Segunda Prova

Nome: ____

5. (Valor: 1,0 ponto) **Para entregar na quinta-feira.** Exercício 4.16 do livro do Manber. Você pode consultar qualquer livro mas não deve discutir com ninguém o exercício.