Estruturas de Dados: MC202

Exercícios

1. A tabela abaixo apresenta três trechos de programas e suas respectivas análises de eficiência, simples e detalhada. Estas análises consideram uma unidade de tempo para qualquer tipo de operação. Verifique estes resultados e justifique os valores correspondentes.

	...
	x = a + b;
	...
		...
		for(i=1; $i \leq n$; i++)
		x = a + b;
		...
			...
			for(i=1; $i \leq n$; i++)
			for(j=1; $j \leq n$; j++)
			x = a + b;
			...

análise simples	1	n	n^2
análise detalhada	2	$5n + 2$	$5n^2 + 5n + 2$

2. Determine quantas vezes é executado o comando de atribuição $x = x + 1$;, no trecho do programa abaixo, em função de n:

```
for (i=1; i<=n; i++)
 for (j=1; j<=i; j++)
   for(k=1; k<=j; k++)
     x = x + 1 ;
```

3. Uma matriz quadrada a é dita diagonal se para todos os valores distintos de i e j tem-se $a[i, j] = 0$. Sugira uma maneira mais eficiente de implementá-la do que a usual.

4. Implemente um tipo abstrato de dados *Polinômio* baseado em listas ligadas que efetue as operações de adição, subtração, multiplicação e divisão de polinômios de grau $n \geq 0$ denotados por:

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x^1 + a_0 x^0,$$

em que $a_n \neq 0$, exceto possivelmente se $n = 0$. Neste caso, são lidos os coeficientes e expoentes de dois polinômios a serem processados (veja o exemplo de soma de polinômios apresentado em sala de aula).

5. Quais as vantagens e desvantagens de se representar um grupo de elementos com um vetor versus uma lista ligada linear?

6. Escreva um programa para executar cada uma das seguintes operações:

   (a) Incluir um elemento no final de uma lista.

   (b) Concatenar duas listas.

   (c) Liberar todos os nós de uma lista.

   (d) Inverter uma lista de modo que o primeiro se torne o último, e assim por diante.

   (e) Eliminar o último elemento de uma lista.

   (f) Eliminar o n-ésimo elemento de uma lista.

   (g) Combinar duas listas ordenadas numa única lista.

   (h) Formar uma lista contendo a união dos elementos de duas listas.

   (i) Formar uma lista contendo a interseção dos elementos de duas listas.

(j) Inserir um elemento após o n-ésimo elemento de uma lista.

(k) Eliminar cada segundo elemento de uma lista.

(l) Colocar os elementos de uma lista em ordem crescente.

(m) Retornar a soma dos inteiros numa lista.

(n) Retornar o número de elementos numa lista.

(o) Deslocar um nó dado p n posições numa lista.

(p) Criar uma segunda cópia de uma lista.

7. Escreva programas para executar cada uma das operações acima sobre elementos armazenados num vetor.

8. Escreva programas para executar cada uma das operações do item 6 supondo que cada lista contenha um nó cabeça com o número de elementos da lista.

9. Escreva programas para executar cada uma das operações do item 6 considerando listas circulares. Quais são mais eficientes com listas circulares do que com listas lineares? Quais são menos eficientes?

10. Escreva uma função que receba uma lista encadeada e retorne o endereço da célula que esteja o mais próximo possível do ponto médio da lista. Isto deve ser feito sem calcular o tamanho n da lista.

11. Aplique às expressões infixas abaixo o algoritmo de conversão para notação posfixa:

```
A*B+C*D^E/F=-G*H
(A+B)*D+E/(F+A*D)+C
(A*(B+(C*(D+(E*(F+G))))))
```

12. Defina uma pilha em termos de um TAD fila.

13. Defina uma fila em termos de um TAD pilha.

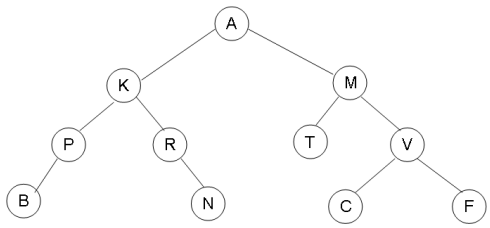
14. A função de Ackermann é definida para valores inteiros e não negativos de m e n da seguinte forma:

$$A(m,n) = \begin{cases} n+1 & se\ \ m = 0 \\ A(m-1,1) & se\ \ m > 0\ \ e\ \ n = 0 \\ A(m-1, A(m,n-1)) & se\ \ m > 0\ \ e\ \ n > 0. \end{cases}$$

Escreva uma função recursiva que recebe as coordenadas m e n de um ponto de uma matriz A, e calcula o valor da função de Ackermann neste ponto.

15. Árvores binárias são uma generalização de listas ligadas. Justifique.

16. Dada a árvore binária da figura abaixo, proponha uma forma de representação da mesma usando uma estrutura de dados matricial.



17. Dê um nome adequado a cada uma das funções XXX, YYY, ZZZ e WWW abaixo:

(a) Função XXX:

```
int XXX (Arvbin *p) {
  if(p != NULL)
    return(1 + XXX(p->esq) + XXX(p->dir)) ;
  else
    return 0 ;
 }
```

(b) Função YYY:

```
int YYY (ArvBin *p) {
 int n,m ;
 if (p == NULL) return 0 ;
 else {
   n = YYY(p->esq) ;
   m = YYY(p->dir) ;
   if (n < m) return (1 + m) ;
   else return (1 + n) ;
 }
```

(c) Função ZZZ:

```
int ZZZ (ArvBin *p) {
  if (p == NULL) return 0 ;
  else
    if ((p->esq == NULL) && (p->dir == NULL))
      return 1 ;
    else
      return (ZZZ(p->esq) + ZZZ(p->dir)) ;
}
```

(d) Função WWW:

```
boolean WWW (ArvBin *p1, ArvBin *p2) {
  if ((p1 == NULL) && (p2 == NULL)) return TRUE ;
  else
    if (p1 == NULL) return FALSE ;
    else
      if (p2 == NULL) return FALSE ;
      else
        if(p1->info == p2->info)
          return ((WWW(p1->esq, p2->esq) && WWW(p1->dir, p2->dir)) ;
        else
          return FALSE ;
 }
```

18. Escreva uma função que encontra um nó com conteúdo k em uma árvore binária.

19. Árvores binárias podem ser usadas de maneira muito natural para representar expressões aritméticas como no exemplo abaixo. Discuta os detalhes desta representação.

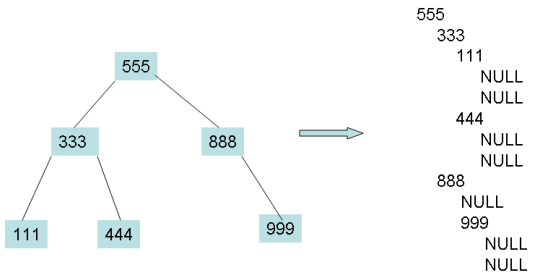
$$((a+b)*c-d)/(e-f)+g$$

20. Escreva uma função que receba uma árvore binária não vazia e devolva o endereço do primeiro nó da árvore percorrida em ordem simétrica (inordem). Escreva outra função que devolva o último nó da árvore. Considere uma versão iterativa e recursiva para este problema.

21. Escreva funções para realizar os três percursos em profundidade de árvores binárias completas e quase completas, representadas sequencialmente. Não use recursão nem estruturas de dados adicionais.

22. Escreva uma função que preencha corretamente todos os campos pai de uma árvore binária.

23. Escreva uma função que imprima o conteúdo de cada nó de uma árvore binária precedido de um recuo em relação à margem esquerda do papel. Esse recuo deve ser proporcional à profundidade do nó (veja exemplo a seguir).



24. Escreva uma função que receba um nó x de uma árvore binária e devolva o nó anterior a x num percurso em inordem.

25. Suponha que nós com chaves $50, 30, 70, 20, 40, 60, 80, 15, 25, 35, 45, 36$ são inseridos, nesta ordem, numa árvore de busca inicialmente vazia. Desenhe a árvore que resulta destas inserções. Em seguida, remova o nó 30 de modo que a árvore continue sendo de busca.

26. Qual a relação existente entre uma árvore de busca e o algoritmo de busca binária implementado a partir de um vetor?

27. Escreva uma função que decida se uma dada árvore binaria é de busca ou não.

28. Considere uma árvore binária de busca sem chaves repetidas. Escreva uma função que receba uma chave k e devolva a chave seguinte em ordem crescente. O que pode acontecer se a árvore possuir chaves repetidas?

29. Escreva uma função que dado o endereço nv de um novo nó criado, insere iterativamente este novo elemento numa árvore binária de busca.

30. Reconstrua a árvore binária a partir dos seguintes percursos em pré-ordem e inordem:

```
Pré-ordem: N J I H G A K F B M L E C D
Inordem:   H I A G J F B K N L C E D M
```