24 de Outubro de 2012

AVALIAÇÃO DE CONHECIMENTO ESPECÍFICO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

01. Toda arvore binária possui as seguintes propriedades:

I. Todos os nós de uma sub-árvore direita são maiores que o nó raiz.
II. Cada Sub-árvore é também uma árvore binária
III. O grau de uma árvore é igual ao máximo dos graus de todos os seus nós
IV. Todos os nós de uma sub-arvore esquerda são menores que o nó raiz
V. Uma árvore binária tem o grau máximo igual a 2

Assinale entre as alternativas abaixo a alternativa correta

a) Apenas os itens I e III são corretos
b) Todos os itens são corretos.
c) Apenas o item V é incorreto
d) Os itens IV e II são corretos
e) Todos os itens são incorretos

02. Seja T uma árvore AVL vazia. Supondo que os elementos 5, 10, 11, 7, 9, 3 e 6 sejam inseridos nessa ordem em T, indique a sequência que corresponde a um percurso de T em pós-ordem.

a) 3, 5, 6, 7, 9, 10 e 11.
b) 7, 5, 3, 6, 10, 9 e 11.
c) 9, 10, 7, 6, 11, 5 e 3.
d) 11, 10, 9, 7, 6, 5 e 3.
e) 3, 6, 5, 9, 11, 10 e 7.

03. Um grafo G(V,E) é uma árvore se G é conexo e acíclico. Assinale entre as definições abaixo aquela que **NÃO pode** ser usada para definir árvores.

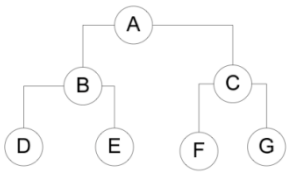
a) G é conexo e o número de arestas é mínimo.
b) G é conexo e o número de vértices excede o número de arestas por uma unidade.
c) G é acíclico e o número de vértices excede o número de arestas por uma unidade.
d) G é acíclico e, para todo par de vértices v, w, que não são adjacentes em G, a adição da aresta (v,w) produz um grafo contendo exatamente um ciclo.
e) G é acíclico e o número de arestas é mínimo.

04. Assinale a alternativa em que todas as propriedades de uma árvore vermelho e preto são verdadeiras.

a) Todo nó é vermelho ou preto. A raiz pode ser vermelha ou preta. Todas as folhas são vermelhas.

b) A raiz é preta. Todas as folhas são vermelhas. Para cada nó, todos os caminhos, desde um nó até as folhas descendentes, contêm um mesmo número de nós pretos.
c) Toda folha (NIL) é preta. Todo nó é vermelho ou preto. A raiz é preta.
d) Se um nó é vermelho, ambos os filhos são vermelhos. A raiz pode ser vermelha ou preta. Todas as folhas são pretas.
e) Todas as folhas são vermelhas. Todo nó é vermelho ou preto. A raiz pode ser vermelha ou preta.

05. Percorrendo a árvore binária a seguir em pré-ordem, obtemos que seqüência de caracteres?.



a) A C G F B E D
b) G C F A E B D
c) A B C D E F G
d) D B E A F C G
e) A B D E C F G

06. Qual a ordem de complexidade da inserção e remoção em uma pilha?

a) O(1) e O(n)
b) O(n) e O(n)
c) O(n) e O(1)
d) O(1) e O(1)
e) Nenhuma das respostas acima.

07. Levando-se em consideração as diferentes estratégias para o gerenciamento da memória nas linguagens de programação (gerenciamento manual x gerenciamento automático), assinale a resposta **correta**:

a) A remoção de um elemento em uma pilha pode ter complexidade O(1) ou O(n) dependendo da linguagem utilizada.
b) A inserção de um elemento na pilha sempre terá complexidade O(n).
c) O esvaziamento completo de uma pilha pode ter complexidade O(1) ou O(n) dependendo da linguagem a ser utilizada, pois isso pode depender da forma de gerenciamento de memória utilizado.
d) A linguagem de programação não afeta a complexidade da inserção, remoção ou esvaziamento completo de uma pilha.
e) Todas as respostas estão erradas.

08. Com relação e Pilhas e Filas podemos afirmar:

a) A primeira utiliza uma estrutura do tipo FILO e a segunda FIFO.
b) A primeira utiliza uma estrutura do tipo FIFO e a segunda FILO.
c) As duas utilizam uma estrutura do tipo FILO.
d) As duas utilizam uma estrutura do tipo FIFO.
e) A forma de implementação é que define a estrutura a ser utilizada por cada uma das estruturas de dados citadas.

09. Suponha o algoritmo descrito no quadro abaixo.

```
Algoritmo Avalia {
  stack = pilha vazia;
  enquanto não terminar a entrada {
      elemento = ler caractere na entrada;
      se o elemento é um operando
          push (stack, elemento);
      senão {
          ele1 = pop(stack);
          ele2 = pop(stack);
          valor = resultado da avaliação da expressão "ele1
elemento ele2";
          push (stack, valor);
      } fim-senão
  } fim-enquanto
  retorne pop(stack);
}
```

Qual será o valor resultante da execução do algoritmo Avalia para a entrada "754+-839/+*"

a) 11
b) 22
c) 30
d) 41
e) Nenhuma das respostas anteriores.

10. Suponha a existência de uma fila inicialmente vazia, com uma variável First apontando para o primeiro elemento e Last apontando para o último elemento. Após a sequencia de operações descritas abaixo, qual alternativa melhor representa o estado atual da fila?

I3 (insere 3), I8, R (remove), I3, I10, I15, I9, I3, R.

a) First -> 3, 10, 15, 9 , 3 <- Last
b) First -> 3, 9, 15, 10 , 3 <- Last
c) First -> 3, 10, 15, 9 <- Last
d) First -> 3, 10, 15, 9 , 3, null <- Last
e) First -> 10, 15, 9 , 3 <- Last

11. É possível implementar uma Fila utilizando uma Pilha?

a) Sim, basta inverter a ordem dos ponteiros internos.
b) Não, filas e pilhas possuem mecanismos diferentes de inserção e remoção. A implementação de um usando outro é basicamente criar o outro elemento, descaracterizando o originalmente utilizado.
c) Sim, é possível alterando a ligação entre os elementos internos da pilha.
d) Sim, é possível utilizado duas pilhas para simular a fila.
e) Sim, basicamente a partir do uso alternado da variável que aponte para o início e fim da pilha.

12. Com relação aos fundamentos e algoritmos das Árvores de Pesquisa, afirma-se:

VI. Seja x um nó em uma árvore de pesquisa binária. Se y é um nó na subárvore esquerda de x, então a chave de y é menor ou igual à chave de x. Se y é um nó na subárvore direita de x, então a chave de x é menor ou igual à chave de y;
VII. As operações de inserção e eliminação provocam mudanças no conjunto dinâmico representado por uma árvore de pesquisa binária. Assim, a estrutura de dados deve ser modificada para refletir essa mudança, mas de tal modo que a propriedade de árvore de pesquisa binária continue válida;
VIII. As árvores B são árvores de pesquisa balanceadas projetadas para funcionar bem em discos magnéticos ou outros dispositivos de armazenamento secundário de acesso direto;
IX. Pesquisar em uma árvore B é semelhante a pesquisar em uma árvore de pesquisa binária, exceto pelo fato de que, em vez de tomar uma decisão de ramificação binária em cada nó, toma-se uma decisão de ramificação de várias vias, de acordo com o número de filhos do nó.

Baseado no exposto, pode-se concluir que:

a) Somente as afirmativas I e II são Verdadeiras;
b) Todas as afirmativas são Falsas;
c) Somente as afirmativas I e III são verdadeiras;
d) Todas as afirmativas são verdadeiras;
e) Somente a Afirmativa III é Falsa.

13. Com relação a Algoritmos, afirma-se:

I. Um algoritmo é dito correto se, para cada instância de entrada, ele pára com a saída correta;
II. A escolha de um algoritmo de ordenação para uma determinada aplicação depende, entre outros fatores, do número de itens a serem ordenados, da extensão em que os itens já estão ordenados de algum modo, de possíveis restrições sobre os valores de itens e da espécie de dispositivo de armazenamento a ser usado: memória principal, discos etc.
III. Algoritmos criados para resolver o mesmo problema caracterizam-se por sempre possuírem a mesma eficiência;

IV. O conjunto de problemas NP-completos tem a propriedade de que, se existe um algoritmo eficiente para qualquer um deles, então existem algoritmos eficientes para todos.

Baseado no exposto, pode-se concluir que:

a) Somente as afirmativas I e II são Verdadeiras;
b) Todas as afirmativas são Falsas;
c) Somente as afirmativas I e III são verdadeiras;
d) Todas as afirmativas são Verdadeiras;
e) Somente a Afirmativa III é Falsa.

14. A Programação Dinâmica e os Algoritmos Gulosos são técnicas importantes para o projeto e análise de algoritmos. Com relação a Programação Dinâmica e Algoritmos Gulosos, **NÃO** se pode afirmar:

a) A Programação Dinâmica aplica-se tipicamente a problemas de otimização em que uma série de escolhas deve ser feita, a fim de alcançar uma solução ótima;
b) A Programação Dinâmica é eficaz quando um dado subproblema pode surgir a partir de mais de um conjunto parcial de escolhas; a técnica chave consiste em armazenar a solução para cada um desses subproblemas, prevendo-se a hipótese de ele reaparecer;
c) A Programação Dinâmica, como o método de dividir e conquistar, resolve problemas combinando as soluções para subproblemas;
d) A Programação Dinâmica só é aplicável quando os subproblemas (resultantes da partição do problema) forem independentes;
e) Um Algoritmo Guloso faz uma escolha ótima para as condições locais, na esperança de que essa escolha leve a uma solução ótima para a situação global.

15. Os grafos são importantes estruturas de dados.Portanto, o conhecimento dos algoritmos para trabalhar com grafos é fundamental. Com relação aos algoritmos de grafos, NÃO se pode afirmar:

a) O algoritmo de Kruskal e o algoritmo de Prim, utilizados para resolver o problema da árvore amplitude mínima, são baseados em abordagem gulosa;
b) O algoritmo de Dijkstra, para resolver o problema de caminhos mais curtos de única origem em um grafo orientado ponderado, utiliza uma estratégia gulosa;
c) Na descrição do tempo de execução de um algoritmo de grafo sobre um determinado grafo $G = (V,E)$, normalmente mede-se o tamanho da entrada em termos do número de vértices $|V|$ e do número de arestas $|E|$ do grafo;
d) No algoritmo da busca (pesquisa) em profundidade, as arestas são exploradas a partir do vértice v mais recentemente descoberto que ainda tem arestas inexploradas saindo dele. Quando todas as arestas de *v* são exploradas, a pesquisa "regressa" para explorar as arestas que deixam o vértice a partir do qual *v* foi descoberto;

16. É **falso** afirmar que.

a) Analisar um algoritmo significa estimar os recursos computacionais requeridos pelo algoritmo;
b) O tempo de execução do pior caso de um algoritmo é um limite superior sobre o tempo de execução para qualquer entrada;
c) Os algoritmos que seguem uma abordagem de dividir e conquistar desmembram o problema em vários subproblemas que são semelhantes ao problema original, mas menores em tamanho, resolvem os subproblemas recursivamente e depois combinam essas soluções com o objetivo de criar uma solução para o problema original;
d) Quando um algoritmo contém uma chamada recursiva a si próprio, seu tempo de execução freqüentemente pode ser descrito por uma equação de recorrência, que descreve o tempo de execução global sobre um problema de tamanho n em termos do tempo de execução sobre entradas menores;
e) Algoritmos de tempo polinomial resolvem problemas em tempo polinomial. Ou seja, esses problemas podem ser resolvidos no tempo O(nk) para alguma constante n, onde k é o tamanho da entrada para o problema.

17. Quais das operações básicas implementadas em listas simplesmente encadeadas são mais custosas computacionalmente quando comparadas às implementações destas operações usando listas sequenciais?

a) Inserção no início e Remoção no início.
b) Inserção no fim e Remoção no fim.
c) Inserção no fim e Remoção no início.
d) Inserção no início e Remoção no fim.
e) Impressão dos elementos.

18. Com base na definição e representação do tipo Lista abaixo, assinale a alternativa que representa corretamente o procedimento de remoção de um elemento de uma lista encadeada. Observe que o procedimento de remoção retorna um ponteiro para Lista, permitindo a atualização após uma eventual remoção do primeiro elemento da mesma.

```
typedef struct lista {
   int info;
   Lista* prox;
} Lista;
```

a)
```
Lista* lst_remove (Lista* l, int v) {
  if (l != NULL) {
    if (l->prox == v) {
      Lista* t = l;
      l = l->info;
      free(t);
    }
    else
      l->prox = lst_remove(l,v);
  }
  return l;
}
```

b)
```
Lista* lst_remove (Lista* l, int v) {
  if (l->info == v) {
```

```
      Lista* t = l;
      l = l->prox;
      free(t);
    }
    return l;
  }
```

c)
```
  Lista* lst_remove (Lista* l, int v) {
    if (l != NULL) {
      if (l->info == v) {
        Lista* t = l;
        l = l->prox;
        free(t);
      }
      else
        l->prox = lst_remove(l->prox,v);
    }
    return l;
  }
```

d)
```
  Lista* lst_remove (Lista* l, int v) {
    if (l->info == v) {
      Lista* t = l;
      l = l->prox;
      free(t);
    }
    else
      l->prox = lst_remove(l->prox,v);
    return l;
  }
```

e)
```
  Lista* lst_remove (Lista* l, int v) {
    if (l != NULL) {
      if (l->info == valor) {
        Lista* t = l;
        l = l->prox;
        free(t);
      }
      else
        lst_remove(l->prox,v);
    }
    return l;
  }
```

19. Uma lista é constituída inicialmente dos seguintes elementos: 20, 11, 36, 73, 08 e 48. Qual é o conteúdo da lista após a execução, em ordem, das seguintes operações.

I. Inserção do elemento 28 no início da lista;
II. Remoção do 3º elemento da lista;
III. Inserção do elemento 18 na 3ª posição da lista;
IV. Remoção do elemento com valor 73;
V. Alteração do elemento da 2ª posição para 05;

a) 28, 05, 18, 11, 08 e 48
b) 05, 08, 18, 28, 36 e 48
c) 28, 20, 05, 18, 08 e 48

d) 28, 05, 36, 18, 08 e 48
e) 28, 05, 18, 36, 08 e 48

20. Leia as afirmativas abaixo sobre Listas e marque a opção **CORRETA**:

I. Os elementos de uma lista encadeada sempre estão dispostos, fisicamente, em posições contínuas de memória;
II. Uma das vantagens das listas sequencias, em relação às listas encadeadas, é a possibilidade de acessar de maneira imediata o elemento de qualquer posição da lista;
III. É possível implementar uma lista encadeada utilizando vetores de estruturas, ao invés de ponteiros. Essa abordagem, porém, não é vantajosa por haver a necessidade de se prever, inicialmente, o tamanho máximo da lista;
IV. Para determinar a quantidade de elementos de uma lista encadeada é necessário percorrer toda a lista para fazer a contagem dos elementos e não existe nenhum mecanismo para obter esse valor diretamente.

a) Apenas II e III estão corretas.
b) Apenas II está correta.
c) Apenas I e IV estão corretas.
d) Apenas II e IV estão corretas.
e) Todas as alternativas estão corretas.

21. Dada as seguintes afirmativas sobre Listas, assinale a alternativa INCORRETA:

a) O procedimento de busca em uma lista ordenada, apesar de ser mais eficiente, tem a mesma complexidade de uma busca em uma lista não ordenada.
b) Uma lista encadeada é dita circular quando o último elemento da lista tem como próximo o primeiro elemento da lista.
c) No procedimento de remoção de um elemento de uma lista duplamente encadeada é sempre necessária a atualização dos nós referentes aos elementos anterior e posterior ao elemento removido.
d) Em uma lista duplamente encadeada e circular, é possível a inserção/remoção imediata do último elemento da lista.
e) A implementação da uma Lista pode ser utilizada para implementação de estruturas com restrições de acesso a elementos, tais como, pilhas e filas.

22. Sobre matrizes, leia as afirmativas abaixo e marque a alternativa **INCORRETA**:

a) Pode-se realizar operações com vetores numéricos da mesma forma como se opera com variáveis numéricas comuns.
b) Um vetor do tipo A e outro de tipo B podem ser representados por uma única matriz de tipo primitivo.
c) Uma matriz de m linhas e n colunas contêm (m * n) dados.
d) Vetor é uma matriz unidimensional

e) O produto de duas matrizes de n linhas e n colunas resulta em uma matriz de n linhas e n colunas.

23. Dada as seguintes afirmativas sobre vetores e matrizes, analise-as e marque a opção **CORRETA**:

I. Uma matriz é uma coleção de variáveis de diferentes tipos, acessíveis com um único nome e armazenados contiguamente na memória.
II. A individualização de cada variável de um vetor é feita através do uso de índices.
III. Em geral, para fazer referência a um valor de um elemento de um vetor, usamos a notação vetor[índice], que serve tanto para obter quanto para definir o valor de um elemento específico, dada sua posição.
IV. A remoção de elementos de um vetor pode ser custosa se não for desejável que existam espaços "vazios" (posições sem valor atribuído) no vetor, pois nesse caso é necessário deslocar o valor de uma posição para outra todos os elementos depois do elemento removido.

a) Apenas II e IV estão corretas.
b) Apenas II e V estão incorretas.
c) Apenas I, III e IV estão corretas.
d) Apenas I e III estão corretas.
e) Apenas II, III e IV estão corretas.

24. O que imprime o programa escrito em "C" abaixo:

```
int f (int a [], int n) {
     if (n <= 0) return 1;
     return a[n-1] * f (a, n-2) + 1;
}

int a [6] = { 0, 1, 2, 3, 4, 5};

#include <stdio.h>

int main() {
     printf ("%d\n", f(a,6));
}
```

a) 35
b) 36
c) 49
d) 79
e) 1957

25. O que imprime o programa escrito em "C" abaixo:

```
#include <stdio.h>

float f (int n, int c, float a[][2]) {
float t=0;
int i;
int j;

for (i=0; i< n; i++)
    for (j=0; j< c; j++)
          t=t+ a[i][j];
return t/(n+c);
}

float mat [3] [2] = {{5.0, 10.0},{20.0, 25.0},{35.0, 40.0}};

int main() {
     printf ("%f\n", f(3, 2, mat));
}
```

a) 45
b) 135
c) 27
d) 70
e) 9

26. Um método que, para ser aplicado a uma estrutura, envolve a aplicação dele mesmo às subestruturas componentes, é chamado de:

a) Auto-relacionamento.
b) Descritor.
c) Ponteiro.
d) Recursividade.
e) Lista.

27. Sobre o conceito de funções, assinale a alternativa **INCORRETA**.

a) A passagem de parâmetros por referência permite que o valor de uma variável passado como argumento seja alterado na função, e sua alteração mantenha-se mesmo após a execução da função.
b) Variáveis locais são declaradas dentro de funções ou procedimentos e são visíveis por outras funções.
c) A implementação iterativa tende a ser ligeiramente mais rápida do que a implementação recursiva já que, em uma implementação recursiva, registra-se o estado atual do processamento de maneira que ela possa continuar de onde parou após a conclusão de cada nova execução subordinada do procedimento recursivo.
d) A ordem da chamada de funções antes ou após a chamada recursiva podem mudar completamente a execução da recursividade.
e) Em uma recursão de cauda, não é necessário guardar a posição onde foi feita a chamada, visto que a chamada é a última operação realizada pela função.

28. Considerando a passagem de informações para funções, julgue os itens abaixo e, em seguida, assinale a opção **CORRETA**.

I. O mecanismo usado para transmitir informações para uma função é denominado argumento.
II. Todos os parâmetros em uma função devem ser do mesmo tipo.
III. Quando a informação é passada por valor, a função chamada não pode alterar o valor de uma variável da função que chama.
IV. A utilização de ponteiros como parâmetros irá causar erro de sintaxe no momento da compilação.

Estão corretas as opções:

a) I e III.
b) I, II, III.
c) II e IV
d) III e IV
e) Todas estão corretas

29. Sobre a programação de computadores considere as seguintes afirmações:

I. Um algoritmo corresponde a uma sequência ordenada de ações que levam à solução de um problema que, quando codificado em uma linguagem de programação, corresponde a um programa de computador.
II. Um algoritmo é dito recursivo quando chama a si mesmo ou chama uma sequência de outros algoritmos, e um deles chama novamente o primeiro algoritmo.
III. Algoritmos recursivos normalmente têm menor tempo de resposta que seus equivalentes iterativos, mas as linguagens PHP e Javascript, por serem linguagens de script, não permitem nem necessitam de recursividade.
IV. Procedimentos e funções são sub-algoritmos codificados como parte do desenvolvimento de um algoritmo para a solução de um problema particular.

Estão **CORRETAS:**

a) Apenas as assertivas II, III e IV estão corretas.
b) Apenas as assertivas I e IV estão corretas.
c) Apenas as assertivas I, II e IV estão corretas.
d) As assertivas I, II, III e IV estão corretas.
e) Todas estão corretas

30. No contexto de da programação de computadores, é correto afirmar sobre a recursividade e a passagem de parâmetros:

a) Para se criar um procedimento é necessário um identificador, uma lista opcional de parâmetros e as ações que nele serão executadas.

b) Um procedimento pode ser ativado em qualquer parte do programa e sua ativação se dá por meio do uso de sua lista de parâmetros.
c) A recursividade é técnica desnecessária caso a programação de um algoritmo seja efetuada em uma linguagem orientada a objetos, uma vez que o envio de uma mensagem a um objeto cria um contexto aninhado que corresponde, indiretamente, à técnica de recursão.
d) Tanto a recursividade direta quanto a indireta não necessitam de uma condição de saída ou de encerramento.
e) Uma característica típica de uma função recursiva é que todos os parâmetros devem ser chamados por valor.