


Pergunta 1

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

O uso de uma lista duplamente ligada (não circular):

Selecione uma opção de resposta:


- ☒ a. Permite que a lista possa ser percorrida em ambas as direções, mas a complexidade das operações de inserção são semelhantes às das lista simplesmente ligadas. ✓
- ☐ b. São menos eficientes que as listas simplesmente ligadas no que toca à inserção de novos elementos, seja à cauda ou à cabeça.
- ☐ c. Permite que a lista possa ser percorrida em ambas as direções, e permite que a inserção à cauda seja mais eficiente que a inserção à cabeça.
- ☐ d. Permite que a lista possa ser percorrida em ambas as direções, e permite que inserção à cabeça e à cauda tenham complexidade $O(1)$

A sua resposta está correta.

Pergunta 2

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

Escolha a única afirmação verdadeira:

Selecione uma opção de resposta:


- ☒ a. É possível aplicar visitas breadth first e depth first a grafos ou a árvores ✓
- ☐ b. As visitas breadth first e depth first só podem ser aplicadas a árvores, sejam estas binárias ou não.
- ☐ c. As visitas breadth first e depth first só podem ser aplicadas a árvores binárias
- ☐ d. As visitas breadth first só pode ser aplicadas a árvores, mas a visita depth first pode ser aplicada também a grafos

A sua resposta está correta.

Pergunta 3

Incorreta

Nota: 0,0 em 0,5

 Marcar pergunta

A atualização do peso de uma incidência num grafo implementado como uma **lista simplesmente ligada de nodos**, com uma **lista simplesmente ligada de incidências**, terá complexidade:

(escolha a que mais se aproximar, considerando n o número de nodos e m o número médio de incidências por nodo)


Selecione uma opção de resposta:

- ☐ a. $O(n + m)$
- ☒ b. $O(n \times m)$ ✗
- ☐ c. $O(n)$
- ☐ d. $O(m)$

Pergunta 4

Correta

Nota: 0.5 em 0.5

 Marcar
pergunta

A definição de uma estrutura para o armazenamento de uma lista duplamente ligada de strings é:

Selecione uma opção de resposta:

☒ a.

```
typedef struct _dlink { struct _dlink* left, *right; char* string; } dLink;
```



☐ b.

```
typedef struct _dlink { struct _dlink previous, next; char* string; } dLink;
```

☐ c.

```
typedef struct _dlink { struct _dlink* left, *right; char string; } dLink;
```

☐ d.


```
typedef struct _dlink { struct _dlink l, r; char string; } dLink;
```

A sua resposta está correta.

Pergunta 5

Respondida

Nota: 0.0 em 2.0

 Marcar
pergunta

Implemente uma função de hash no intervalo $[0, MAX[$ com a seguinte assinatura:

```
#define MAX 100000
```


```
int hash(char* string) { ... }
```

```
return atoi(string)%MAX;
```

Pergunta 6

Incorreta

Nota: 0,0 em 0,5

 Marcar
pergunta

Qual das seguintes operações de inserção é a menos eficiente quando se pretende apenas o armazenamento de nova informação?

Selecione uma opção de resposta:


- ☐ a. Inserção à cauda numa lista ligada simples
- ☒ b. Inserção ordenada numa lista duplamente ligada ✗
- ☐ c. Inserção à cabeça numa lista ligada simples
- ☐ d. Inserção em árvores binárias de procura

A sua resposta está incorreta.

Pergunta 7

Incorreta

Nota: 0,0 em 0,5

 Marcar
pergunta

Numa lista simplesmente ligada com n elementos, a inserção à cabeça tem complexidade:

Selecione uma opção de resposta:


- ☐ a. $O(1)$
- ☒ b. $O(n)$ ✗ $O(n^2)$

A sua resposta está incorreta.

Pergunta 8

Respondida

Nota: 0,0 em 4,0

 Marcar
pergunta


Defina a estrutura de dados necessária para o armazenamento de um grafo orientado e pesado, em que os nodos são valores inteiros, e os pesos são valores reais, usando listas de incidências. Implemente um método que retorna a aresta com peso menor entre 2 vértices, tendo em consideração que podem existir mais que uma aresta entre cada par de vértices.

```
typedef struct grafo {
    Node* node;
    Node* connection;
    float cost;
} Grafo;
```

Pergunta 9

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

O algoritmo de Dijkstra permite:

Selecione uma opção de resposta:


- ☐ a. Encontrar o caminho mais longo entre dois nodos de um grafo
- ☐ b. Encontrar o caminho mais longo entre dois nodos de uma árvore
- ☐ c. Encontrar o caminho mais curto entre dois nodos de uma árvore
- ☒ d. Encontrar o caminho mais curto entre dois nodos de um grafo ✓

A sua resposta está correta.

Pergunta 10

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

A travessia de uma árvore pode ser realizada de diferentes formas. Selecione a única afirmação verdadeira:

Selecione uma opção de resposta:


- ☐ a. As visitas breadth first recorrem ao uso de uma queue, enquanto que as visitas depth first recorrem a uma stack
- ☐ b. As visitas breadth first recorrem ao uso de uma stack, enquanto que as visitas depth first recorrem ao uso de uma queue
- ☐ c. É possível implementar visitas breadth first ou depth first sem o uso de estruturas de dados auxiliares
- ☒ d. Nenhuma das outras afirmações é verdadeira ✓

A sua resposta está correta.

Pergunta 11

Respondida

Nota: 1,0 em 2,0

 Marcar pergunta


Implemente uma função que receba uma lista simplesmente ligada de palavras (strings) e devolva o número de palavras que **começam e acabam** com a letra 'a'.

```
int visitar_lista ( Lista *lst)
{
    int conta = '0';
    List *palavra;
    for (palavra = lst; palavra; palavra = palavra->next)
    {
        char* palavra;
        if (palavra[0]=='a' && palavra[strlen(palavra)-1]=='a') conta++;
    }
    return conta;
}
```

Pergunta 12

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

Escolha a afirmação que lhe parecer mais completa:

Selecione uma opção de resposta:


- ☒ a. Uma função de hash retorna um inteiro no intervalo $[0, m[$ com m definido pelo programador. ✓
- ☐ b. Uma função de hash retorna um inteiro
- ☐ c. Uma função de hash retorna um inteiro positivo
- ☐ d. Uma função de hash retorna um inteiro no intervalo $[1, 100]$

A sua resposta está correta.

Pergunta 13

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

Comparando a estrutura de uma árvore binária de procura com uma lista duplamente ligada.

Selecione uma opção de resposta:


- ☐ a. As estruturas não são comparáveis.
- ☒ b. As estruturas são equivalentes. ✓
- ☐ c. A árvore binária de procura precisa tem uma estrutura mais complexa que lista duplamente ligada.
- ☐ d. A árvore binária de procura precisa tem uma estrutura menos complexa que lista duplamente ligada.

A sua resposta está correta.

Pergunta 14

Correta

Nota: 0,5 em 0,5

 Marcar pergunta

Seria possível implementar um grafo orientado e pesado com recurso a árvores binárias de procura?

Selecione uma opção de resposta:


- ☐ a. Nos grafos podemos usar árvores binárias de procura para os nodos (vértices de partida), mas não podemos usar árvores para as incidências (vértices de chegada).
- ☐ b. Nos grafos apenas podemos usar listas ligadas simples para os nodos (vértices de partida), mas podemos usar árvores binárias de procura para as incidências (vértices de chegada).
- ☐ c. Não faz sentido usar árvores binárias de procura em grafos porque não existem problemas com um número elevado de dados que o justifique.
- ☒ d. Sim, num grafo podemos substituir a listas de nodos e as listas de incidências por árvores binárias de procura. ✓

A sua resposta está correta.

Pergunta 15

Correta

Nota: 0,5 em 0,5

 Marcar
pergunta

Considere a seguinte árvore:



A sequência: Q, E, F, G, J, C, B, A, D, N, K, Z corresponde a uma visita:

Selecione uma opção de resposta:


- ☒ a. Depth first, Pre order ✓
- ☐ b. Depth first, Post order
- ☐ c. Depth first, In order
- ☐ d. Breadth First

A sua resposta está correta.

Pergunta 16

Incorreta

Nota: 0,0 em 0,5

 Marcar
pergunta

Numa árvore binária de procura, quanto à complexidade da operação de inserção, qual o pior cenário.

Selecione uma opção de resposta:

- ☒ a. A inserção numa árvore binária de procura tem sempre complexidade $O(\log_2 n)$ em qualquer cenário. ✗
- ☐ b. Quando a lista de valores a inserir já está ordenada de forma crescente por ser uma inserção à cauda.
- ☐ c. Quando a lista de valores a inserir já está ordenada de forma decrescente por ser uma inserção à cauda.
- ☐ d. Quando a lista de valores já se encontra ordenada.

Pergunta 17

Incorreta

Nota: 0,0 em 0,5

☐ Marcar
pergunta

Qual das seguintes operações de inserção é mais eficiente quando se pretende apenas o armazenamento da nova informação?

Selecione uma opção de resposta:

- ☐ a. Inserção ordenada numa lista duplamente ligada
- ☒ b. Inserção à cauda numa lista ligada simples ✖
- ☐ c. Inserção à cabeça numa lista ligada simples
- ☐ d. Inserção em árvores binárias de procura

A sua resposta está incorreta.

Pergunta 18

Correta

Nota: 0,5 em 0,5

☐ Marcar
pergunta

As listas circulares:

Selecione uma opção de resposta:

- ☒ a. Permitem a inserção à cabeça e à cauda com complexidade $O(1)$ ✔
- ☐ b. Permitem a inserção à cabeça, à cauda, e ordenada, com complexidade $O(1)$
- ☐ c. Permitem a inserção à cabeça e à cauda com complexidade $O(1)$, mas pioram a complexidade da inserção ordenada para $O(n^2)$ por ser necessário percorrer a lista em ambos os sentidos.
- ☐ d. Têm complexidade $O(n)$ para qualquer operação de inserção.

A sua resposta está correta.

Pergunta 19

Incorreta

Nota: 0,0 em 0,5

☐ Marcar
pergunta

A utilização de uma matriz de adjacências para representação de um grafo não pesado e não direcionado:

Selecione uma opção de resposta:


- ☐ a. Não é recomendável a sua utilização por haver um grande desperdício de memória e a necessidade saber de antemão a quantidade total de nodos ou realocar memória e reorganizar a matriz.
- ☒ b. Não é recomendável porque a complexidade de uma pesquisa é $O(n^2)$. ✖
- ☐ c. A matriz de adjacências só pode ser utilizada para grafos orientados.
- ☐ d. Pode ser utilizada desde que haja uma garantia de não haver mais de 2 milhões de vértices.

A sua resposta está incorreta.

Pergunta 20

Respondida

Nota: 2,0 em 2,0

 Marcar pergunta

Considerando a estrutura típica de uma árvore binária de procura, escreva uma função que calcule a sua altura (altura máxima).


```
int altura (ABin a)
{
    int r=0,altesq=0,altdir=0;
    if (a){
        altesq = altura (a->esq);
        altdir = altura (a->dir);
        r = 1 + max (altesq,altdir);
    }
    return r;
}
```

Comentário:

Pergunta 21

Respondida

Nota: 0,0 em 2,0

 Marcar pergunta

Implemente uma função que receba uma lista simplesmente ligada de inteiros e a devolva por ordem inversa.

```
void inverter(struct ListaInteiros **lst) {
    struct ListaInteiros *buffer = NULL;
    struct ListaInteiros *inicio = *lst;

    while (inicio != NULL) {
        struct ListaInteiros *temp = head->next;
        head->next = buff;
        buffer = inicio;
        inicio = tempo;
    }
    *lst = buffer;
}
```