Universidade Federal de Santa Catarina - Campus Blumenau
Departamento de Engenharia de Controle, Automação e Computação **Recuperação**
BLU3202 — **Algoritmos e Estruturas de Dados** 2022/1

Aluno(a): .....

1. Sejam três algoritmos A_1, A_2 e A_3, cujas complexidades podem ser dadas, respectivamente, pelas expressões $C_{A_1}(n) = 3n^2 + 2n + 7$, $C_{A_2}(n) = 30n + 5$ e $C_{A_3}(n) = 5log_2 n + 2$. Nas alternativas abaixo, assinale V nas que forem verdadeiras e F nas que forem falsas.

   ( ) Existe pelo menos um tamanho de entrada $n > 0$ para a qual, no pior caso, A_1 é mais rápido que A_2

   ( ) A_2 será, no pior caso, mais rápido que A_1 independente do tamanho da entrada

   ( ) Os três algoritmos levarão mais que o dobro do tempo para processar uma entrada de tamanho $2n$ em comparação com uma entrada de tamanho n (para qualquer valor de n)

   ( ) Existe um valor x tal que, para qualquer entrada de tamanho $n > x$, A_3 será mais rápido que A_1

   ( ) Existe um valor x tal que, para qualquer entrada de tamanho $n > x$, A_2 será mais rápido que A_3

   ( ) Todos os algoritmos tem complexidade superior à da inserção no início de uma lista encadeada

   ( ) Todos os algoritmos tem complexidade superior à da inserção no fim de uma lista encadeada

2. O que será impresso pelo programa abaixo?

```
1     void k1(int p, int q){
2        int r;
3        r=p; p=q; q=r;
4     }
5
6     void k2(int *v, int *w){
7        int z;
8        z=*v; *v=*w; *w=z;
9     }
10
11    int main(){
12       int a=58, b=60, c=1;
13       k1(a,b);
14       k2(\&b, \&c);
15       printf("\%d",c-a-b);
16    }
```

   Resposta: _____

3. Quantas chamadas recursivas a função abaixo faz quando `n=5`, (excluindo a primeira chamada à função)?

```
int f(int n){
  if (n==1) return n;
     return n*f(n-1);
}
```

   Resposta: _____

4. Seja uma lista duplamente encadeada que contem n elementos. Qual será a quantidade de memória ocupada apenas pelos ponteiros desta lista considerando-se que cada ponteiro ocupa p bytes? Escolha uma opção:

   ( ) np bytes

   ( ) $4np$ bytes

   ( ) $6np$ bytes

   ( ) $2np$ bytes

   ( ) np^2 bytes

5. Considere as afirmativas a seguir a respeito de filas, pilhas e listas. Assinale V para as alternativas verdadeiras e F para as falsas.

   ( ) As estruturas de dados pilhas, filas e listas armazenam coleções de itens. A característica que as distingue é a ordem em que podem ser retirados os itens dessas coleções em relação à ordem em que foram inseridos.

   ( ) Considere que os itens A, B, C, D, E foram inseridos nessa ordem em uma fila. Necessariamente, o primeiro elemento a ser removido dessa fila é o elemento A.

   ( ) Considere que os itens A, B, C, D, E foram inseridos nessa ordem em uma pilha. Necessariamente, o último elemento a ser removido dessa pilha é o elemento E.

   ( ) Considere que os itens A, B, C, D, E foram inseridos nessa ordem em uma lista. Necessariamente, o primeiro elemento a ser removido dessa lista é o elemento A.

6. Considere a inserção das seguintes chaves na dada ordem em uma tabela hash de tamanho 7: {**290, 37, 252, 27, 125, 267, 271**} usando o método da divisão com a função hash $\mathbf{h(x) = x \bmod 7}$. Mostre as tabelas resultantes resolvendo as colisões por encadeamento e por sondagem linear. Responda no seguinte formato: `0:[...], 1:[...], ...., n:[...]`

   _____

   _____

   _____

   _____

   _____

   _____

   _____

   _____

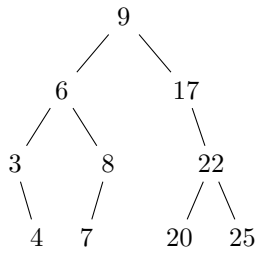
   _____

7. Escreva abaixo a sequência de números que será impressa caso a árvore seja percorrida

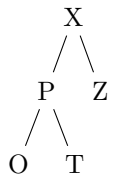
   a) Em pré-ordem: _____

   b) Em ordem: _____

   c) Em pós-ordem: _____



8. Seja uma árvore binária de pesquisa que armazena caracteres em ordem alfabética. Se essa árvore estiver vazia, a inserção, em sequência, dos caracteres **X**, **P**, **T**, **Z** e **O** produzirá o seguinte resultado:



   Considere-se, em vez disso, a inserção, nessa mesma árvore vazia, da seguinte sequência de caracteres: **C O N T R L E A U M**.

   Preencha corretamente as afirmações sobre a árvore resultante dessa inserçaõ:

   - A altura da árvore será _____ .
   - A raiz da árvore armazenará a letra _____.
   - A árvore tem _____ folhas.
   - A altura da árvore é _____.
   - A profundidade do nó que armazena a letra **L** é _____.
   - Enquanto o caminhamento *em ordem* sobre essa árvore produzirá a cadeia de caracteres ACELMNORTU, o caminhamento em *pré-ordem* produzirá a cadeia de caracteres _____ e o caminhamento em pós-ordem produzirá a cadeia _____ .

9. Considere as afirmações a seguir a respeito de pesquisa binária e sequencial. Assinale V para as afirmações verdadeiras e F para as falsas.

   ( ) Em um vetor com n elementos, no pior caso, a pesquisa binária avalia log2n elementos.

   ( ) No pior caso, a pesquisa binária avalia metade dos elementos do vetor.

   ( ) A pesquisa sequencial requer que os elementos do vetor estejam ordenados.

   ( ) No pior caso, a pesquisa sequencial avalia todos os elementos do vetor.

   ( ) A pesquisa binária pode ser feita sobre vetores em que os elementos estejam desordenados.

10. Sejam os algoritmos de ordenação enumerados a seguir:

    (a) *Insertion sort*

    (b) *Selection sort*

    (c) *Merge sort*

    (d) *Quick sort*

    Numere as alternativas a seguir conforme os algoritmos de ordenação mencionados acima. Algumas alternativas podem ter mais de uma resposta correta.

    ( ) Cada um dos elementos da porção não ordenada do vetor deve ser colocado na posição adequada dentro da porção ordenada. Repete-se esse procedimento do segundo ao último elemento.

    ( ) Divide-se o vetor ao meio, repete-se a divisão para cada um dos subvetores, até que cada subvetores tenha apenas 1 elemento. Nesse ponto, faz-se o reagrupamento dos subvetores de forma que eles fiquem ordenados. Repete-se este procedimento até restar um só grupo de elementos.

    ( ) Escolhe-se um ponto de referência (pivô) e separam-se os elementos em 2 partes: à esquerda, ficam os elementos menores que o pivô, e à direita, os maiores. Repete-se este processo para os grupos de elementos formados (esquerda e direita) até que todos os elementos estejam ordenados.

    ( ) Encontra o menor elemento e o troca com o elemento que ocupa a primeira posição, depois o segundo menor com a segunda posição e assim sucessivamente (n-1 vezes).

    ( ) Tem, no melhor caso, a mesma complexidade do *merge sort*.

    ( ) Tem, no pior caso, complexidade $O(n.log_2 n)$.

    ( ) Tem, no pior caso, complexidade $O(n^2)$.