Ministério da Educação
Universidade Federal de Santa Maria
Campus de Frederico Westphalen-RS
Departamento de Tecnologia da Informação
Curso de Sistemas de Informação

Avaliação da Disciplina (Trabalho 02) _ Peso 03 pontos
Disciplina: Pesquisa e Ordenação de Dados **Data:**
Aluno: MAURÍCIO WITTER Nota: _____

1. Com relação aos conceitos de dados, informação e conhecimento, avalie as afirmações seguintes: (01 ponto)
   I. Conhecimento é estar ciente e ter um entendimento de um conjunto de informações. É saber como essas informações podem ser úteis para suportar determinado processo ou tarefa. É o esforço de investigação para descobrir aquilo que está oculto e que ainda não está compreendido. Adquirir conhecimento não é reter informação, mas utilizá-la para descobrir novidades e evoluir permanentemente.
   II. Dados são informações que foram convertidas em contexto significativo e útil para os usuários, fornecendo subsídios para compreender uma situação, resolver problemas e tomar decisões. Portanto, dados são informações que foram processadas, relacionadas, tratadas e contextualizadas.
   III. Dado é a simples representação de um determinado fato, sem que se possa estabelecer algum significado. É um simples registro, um elemento bruto que, isoladamente, não possibilita análise e não conduz a uma compreensão de determinado fato ou situação.
   IV. Dado processado sempre gera informação e informação sempre gera conhecimento.
   São falsas as afirmações:
   a( ) II E III.
   b( ) I E IV.
   c( ) I E III.
   d(X) II E IV.
   e() I, II E III.

2. O trabalho nas organizações foi transformado pelos sistemas de informação e, sem dúvida, seria muito difícil administrar organizações hoje em dia sem fazer uso desses sistemas, mesmo no caso das pequenas empresas. Um grande número de administradores já usa sistemas e tecnologias de informação diariamente para cumprir suas tarefas, desde ferramentas de produtividade até aplicações para coordenar o negócio como um todo. No que diz respeito aos conceitos relacionados à sistema de informação, avalie as informações a seguir: (01 ponto)
   I. Conhecimento quer dizer informações apresentadas em uma forma significativa e útil para os seres humanos.
   II. Informação quer dizer dados apresentados em uma forma significativa e útil para os seres humanos.
   III. Conhecimento é um entendimento ou modelo, sobre pessoas, objetos e eventos, derivado de informações sobre eles.
   IV. Informações são sequências de fatos ainda não analisados.
   V. Dados são sequências de fatos ainda não analisados.
   É correto apenas o que se afirma em
   a( ) I, II e III.
   b(X) I, III e V
   c( ) III, IV e V.
   d( ) II, III e V.
   e( ) I, II, IV e V.

3. No contexto de ordenação de dados, analise as afirmações a seguir e marque V para as afirmações verdadeiras e F para as afirmações falsas: (01 ponto) *Obs: A questão será considerada correta apenas se todas as alternativas forem assinaladas de forma correta;*
   (V) Em uma ordenação *in-place*, não é utilizado espaço adicional durante o processo;
   (V) Ordenação Interna é aquela realizada utilizando apenas o recurso de memória principal e geralmente aplicada em volumes pequenos de dados;
   (V) Ordenação externa é aquela aplicada a grandes volumes de dados é realizada utilizando memória secundária;

4. No contexto dos algoritmos de ordenação de dados, explique, com exemplos, o significado dos termos *Ordenação Estável* e *Ordenação Não Estável (02 pontos)*

A ordenação estável recebe um array de input e o output da ordenação deverá ter os elementos duplicados na mesma posição relativamente, ou seja, o elemento duplicado que estiver mais à direita, deve permanecer mais à direita depois da ordenação do que o seu clone. Por exemplo:

Estável:
I = [5, 2, 6, **2**];
O = [2, **2**, 5, 6];

Não estável:
I = [5, 2, 6, **2**];
O = [**2**, 2, 5, 6];

A ordenação instável troca os elementos duplicados de posição, o que torna o algoritmo instável pois não é necessário essa troca. Os algoritmos do paradigma dividir e conquistar como Heap, apesar de ser instável, ainda realiza a ordenação em um tempo de complexidade menor do que um algoritmo de ordenação estável. Nesse sentido, cabe avaliar a cada caso qual tem os melhores prós a ordenação em si.

5. A tabela a seguir exibe as implementações, em Java, de alguns dos algoritmos de ordenação estudados em sala de aula. Analise cada um deles e preencha a coluna da direita. (05 pontos)

1) Implementação do Algoritmo na Linguagem de Programação Java	**2) Identificação e Características (Coloque aqui o nome do algoritmo utilizado na implementação, suas características, aplicações, forma de funcionamento, vantagens e/ou desvantagens.**

`static void Ordena(int[] v, int n)`
`{`
`  int i,j,menor_idx,aux;`
`  for (i = 0; i < n-1; i++)`
`  {`
`  menor_idx = i;`
`  for (j = i+1; j < n; j++)`
`  if (v[j] < v[menor_idx]) {`
`  menor_idx = j;`
`  }`
`  aux = v[menor_idx];`
`  v[menor_idx] = v[i];`
`  v[i] = aux;`
`  }`
`}`
	Title: **Selection Sort**
	- Seleciona o menor elemento da lista não ordenada a cada iteração e o coloca na sua devida posição no início, depois do menor elemento já ordenado, considerando a ordem crescente, caso contrário seleciona o maior;
	- Ex: [20, 12, 7, 5]: => [5, 12, 7, 20] => [5, 7, 12, 20];
	- Simples de ser implementado;
	- In-place: não requer array auxiliar
	- Não estável: elementos iguais são trocado de posição;
	- Verificando os dois loops e a condicional, temos que ele será O(n²) em todos os casos de complexidade, pois n*n == n²;
	- Espaço de complexidade: O(1).
	
	Aplicações:
	- O Selection Sort pode ser utilizado para gravação de dados em memórias flash, visto que essas memórias têm sua vida útil diminuída pelas gravações por conta do seu número de trocas se O(n). Além disso, ele é preferível ao algoritmo Bubble Sort nesse caso que tem O(n²) de trocas.
	- Ordenar pequenas listas de dados;
	
	Vantagens:
	- Como no pior caso ele é O(n²) com O(n) trocas, pode ser uma boa solução caso o custo de memória seja alto;
	- Simples de ser implementado;
	
	Desvantagens:
	- Levando em consideração o tempo de classificação, ele é lento para grandes conjuntos de dados;
	- Não estável;
	- Complexidade de tempo quadrática.

Ministério da Educação
Universidade Federal de Santa Maria
Campus de Frederico Westphalen-RS
Departamento de Tecnologia da Informação
Curso de Sistemas de Informação

```
static void Ordena(int[] v, int n)
{
    for (int i=1; i<n; ++i)
    {
        int key = v[i];
        int j = i-1;
        while (j>=0 && v[j] > key)
        {
            v[j + 1] = v[j];
            j = j-1;
        }
        v[j+1] = key;
    }

}
```

Title: **Insertion Sort**

- O primeiro elemento inserido já está ordenado a princípio, ao adicionar o segundo elemento, se o segundo não está ordenado e for maior que o anterior, colocasse ele à direita, caso contrário à esquerda; caso ele seja menor que o elemento[0] trocasse o segundo elemento com o elemento do primeiro índice. Assim sucessivamente.

- Armazenamos o elemento do índice atual (index 1);
- Pegamos o índice menor que o atual (index 0);
- Comparamos se o elemento do menor índice é maior que o elemento do índice atual, ex: el[0] > aux[1];
- Se for, colocamos o elemento do menor índice no menor índice + 1, ou seja, no índice da frente;
- Descontamos 1 do índice menor;
- Loop passa novamente, se a condição for atendida, ordenando os elementos menores que já foram trocados.
- Colocamos o elemento do índice atual no índice anterior.

Ex: [9, 5, 1] > [9, 9, 1] > [5, 9, 1] > [5, 9, 9] > [5, 1, 9] > [5, 5, 9] > [1, 5, 9]

Aplicações:
- Ordenar pequenas listas de dados;
- Demonstrações didáticas.

Vantagens:
- Estável: não altera a ordem relativa dos elementos duplicados;
- In-place: necessita apenas de uma variável constante;
- Ordena a lista à medida que insere;
- Simples de ser implementado;
- Eficiente para pequenos conjuntos de dados;
- Preferível ao Selection Sort e Bubble Sort.
- No melhor caso é O(n) com O(1) trocas

Desvantagens:
- No pior caso e no caso médio será O(n^2);
- Realiza O(n^2) trocas;

```
static void Ordena(int[] v, int n)
   {
       int i, j,aux;
       for(i=n-1; i>0; i--){
        for(j=0; j<i; j++){
            imprimeValores(v);
           if( v[j]> v[j+1]){
               aux = v[j];
               v[j] = v[j+1];
               v[j+1] = aux;
           }
        }
     }
   }
```

title: **Bubble Sort**

- In-place: não necessita de array auxiliar O(n);
- Simples, ordena trocando valores adjacentes;
- A ordenação funciona iterando sobre o array duas vezes, uma irá percorrer todo array trocando os elementos de posição, e a outra irá percorrer n-1 vezes fazendo com que a iteração interna repita esse processo até estar completamente ordenado. Dessa forma, comparando se o elemento anterior é maior que o elemento posterior ao anterior, se for, aux recebe o maior valor, o array no índice atual recebe o elemento do índice posterior a ele, ou seja, o menor que ele, o array no índice atual + 1 recebe o maior elemento;
- Não estável: troca os elemento duplicados relativos a entrada de posição;
- No pior caso será O(n²) e no melhor caso será O(n);
- Complexidade espacial O(1).

Ex:
[**7**, **2**, 9, 1] > [**2**, **7**, **9**, 1] > [2, 7, **9**, **1**] > [2, 7, **1**, **9**];
[**2**, **7**, 1, 9] > [2, **7**, **1**, 9] > [2, ***1***, **7**, **9**] > [2, 1, **7**, **9**];
[**2**, **1**, 7, 9] > [***1***, **2**, **7**, 9] > [1, 2, **7**, **9**] > [1, 2, 7, 9];

Aplicações:
- Ordenar pequenas listas de dados
- Pode ser usado para encontrar pequenos erros na computação gráfica;
- Usado para algoritmos de preenchimento de polígonos, visto que compara elementos adjacentes, tal como x e y

Vantagens:
- Simples de ser implementado, com pouco código;

Desvantagens:
- Lento, no pior caso O(n²) para listas ordenadas de forma inversa e em caso de uma lista de dados grande.
- Tem O(n²) de trocas, ou seja, gasta recursos de memória como as memórias flash;
- Realiza muitas operações e comparações;