



UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO
Rua Dom Manoel de Medeiros, s/n – Dois Irmãos 52171-900 Recife-PE
Fone: 81 3302 1000 www.dc.ufrpe.br

DISCIPLINA: Algoritmos e Estruturas de Dados	CÓDIGO: 06214
DEPARTAMENTO: Computação	ÁREA: Informática
CURSO: Licenciatura em Computação	
PROFESSOR RESPONSÁVEL: Luciano Demétrio Santos Pacífico	
DATA MÁXIMA DE ENTREGA: 29-08-2020	

Regras da Lista de Exercícios 02

1. **Não é permitido o uso de Estruturas de Dados prontas de Linguagens de Programação.** O aluno deve implementar suas próprias Estruturas de Dados. Na Linguagem de Programação **C**, deve-se usar **structs**. Nas demais Linguagens de Programação permitidas (vide **Regras da Disciplina**), deve-se usar **classes**.
2. **Não é permitido o uso de Algoritmos e comandos otimizados prontos de Linguagens de Programação.** Todos os algoritmos solicitados devem ser implementados pelos alunos como **procedimentos** (funções, métodos, etc.).
3. As questões que solicitam **escrita de código** devem ser resolvidas **apenas através dos recursos oferecidos pela pseudolingagem definida para a disciplina, e dos recursos equivalentes em Linguagens de Programação reais**, sendo eles: variáveis, constantes e tipos primitivos, expressões, estruturas condicionais, estruturas de repetição, sub-rotinas, estruturas de dados homogêneas (Arrays) e estruturas de dados heterogêneas (registros – classes e structs).
4. Para a Lista de Exercícios 02, os **Arrays** devem ser implementados através de **Alocação Estática**. Não será permitido o uso de **Estruturas de Dados dinâmicas** implementadas em Linguagens de Programação, como os **Vectors** e **Lists** da Linguagem de Programação **Java**, por exemplo. Deve-se usar **arrays**.
5. Como a Linguagem de Programação **Python** não suporta **arrays estáticos**, o aluno que optar por usar esta Linguagem deverá usar **Lists**, **única e exclusivamente para simular o comportamento de arrays estáticos**, de acordo com os seguintes critérios:
 - a. Deve-se criar uma **List** com posições vazias através do método **append** dessa estrutura. O uso do **append** será permitido **apenas na alocação de memória para a variável que representará o array estático**;
 - b. O limite máximo de **M** objetos deve ser controlado **através de código**;
 - c. A Estrutura deve ser manipulada como se fosse um **array estático**, com os **procedimentos escritos pelo aluno, não sendo permitido** o uso de métodos, funções ou otimizações oferecidos pela Linguagem **Python**.
6. **Regra de Ouro:** Todos os alunos envolvidos em **cópias** terão suas notas **ANULADAS** nas referidas questões.
7. Apenas o código “.c”, “.cpp”, “.java”, “.py”, etc. deve ser enviado ao professor para cada questão. Deve-se enviar **um único arquivo resposta por questão**, que conterá todas as classes/estruturas e procedimentos necessários para a solução da questão. Todos os arquivos devem ser enviados **em uma única pasta, “zipados”**.
8. O arquivo de resposta com o código para cada questão deve ser **nomeado** na forma “L#Q%.c”, “L#Q%.java”, etc., onde “#” refere-se ao número da lista e “%” refere-se ao número da questão (ex.: L1Q2.c, para o arquivo de resposta da segunda questão da primeira lista, usando a Linguagem C).
9. As respostas da Lista de Exercícios 02 devem ser submetidas **unicamente através da tarefa criada no Google Classroom para este propósito**.

Lista de Exercícios 02 – Listas Lineares Sequenciais

1. Escreva os algoritmos, usando uma **Linguagem de Programação**, de **Visualização (Impressão)**, **Busca**, **Inserção** e **Remoção** em uma **Lista Linear Sequencial Ordenada** com capacidade máxima para **M** elementos, de acordo com o especificado: (6.0 pontos)
 - a. Defina as Estruturas de Dados básicas necessárias.
 - b. O tamanho **M** deve ser definido pelo usuário no início do uso da aplicação (parâmetro).
 - c. A lista deverá estar inicialmente vazia.
 - d. Na operação de impressão, posições não ocupadas da estrutura devem ser impressas como **NIL**.
 - e. A lista deverá se manter ordenada após as operações de **inserção** e **remoção**.
 - f. Deve haver uma forma de **interação do usuário** com o programa, permitindo que esse usuário execute as operações de visualização, busca, inserção e remoção, **de acordo com a sua necessidade**.
 - g. Os procedimentos devem imprimir **mensagens de erro**, caso o usuário tente realizar **operações inválidas** (com remoções de dados inexistentes na lista, ou inserção de mais elementos do que o suportado pela Estrutura de Dados).
2. Escreva os algoritmos, usando **uma Linguagem de Programação**, de **Visualização (Impressão)**, **Busca**, **Inserção** e **Remoção** em uma **Lista Linear Sequencial** com capacidade máxima **inicial** para **M** elementos, de acordo com o especificado: (4.0 pontos)
 - a. Defina as Estruturas de Dados básicas necessárias.
 - b. O tamanho **M** inicial deve ser definido pelo usuário no início do uso da aplicação (parâmetro).
 - c. A lista deverá estar inicialmente vazia.
 - d. Na operação de impressão, posições não ocupadas da estrutura devem ser impressas como **NIL**.
 - e. A operação de **inserção** deve ser implementada de modo que, **sempre** que uma tentativa de inserção for realizada quando a lista **já contiver M elementos**, um novo array estático deve ser criado, com **capacidade máxima $2 \cdot M$** , e todos os elementos contidos no array original **devem ser copiados** para o novo array. Após a cópia, o array original deve ser **substituído** pelo novo, e o tamanho da lista nova deve ser **atualizado e impresso para o usuário**.
 - f. Deve haver uma forma de **interação do usuário** com o programa, permitindo que esse usuário execute as operações de visualização, busca, inserção e remoção, **de acordo com a sua necessidade**.
 - g. Os procedimentos devem imprimir **mensagens de erro**, caso o usuário tente realizar **operações inválidas** (com remoções de dados inexistentes na lista).