UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Faculdade da Computação
2º Trabalho de Algoritmos e Estrutura de Dados 1
Prof. Luiz Gustavo Almeida Martins

- Os códigos deverão ser implementados somente em Linguagem C, sendo necessária a utilização das estruturas de dados conforme discutido nas aulas.
- Deve-se aproveitar o conhecimento sobre a estrutura para buscar a maior eficiência das operações implementadas no TAD.
- A entrega do trabalho consiste no envio de um arquivo zip, contendo todos os arquivos necessários para a compilação e execução do programa, organizados em diretórios (uma pasta por questão). O envio é individual e deve ser feito até a data e horário especificados na tarefa no ambiente virtual. A integridade desse arquivo é de responsabilidade dos alunos.
- A apresentação do trabalho será individual em data e horário previamente agendados entre o professor e o grupo. O representante do grupo deve entrar em contato com o professor através do chat para o agendamento.

1) Implemente um TAD Pilha de **números inteiros** usando alocação **estática/sequencial** com todas as operações básicas e a operação ***esvazia_pilha***: que recebe o endereço de uma pilha e a retorna para o estado de vazia.

Utilizando esse TAD, desenvolver um programa aplicativo que faça a conversão de um número inteiro positivo na base 10 (digitado pelo usuário), incluindo o zero, para outras bases de acordo com a opção escolhida pelo usuário (B-binário, O-octal e H-hexadecimal). O programa deve repetir esse processo até que o usuário digite um número negativo.

2) Implemente um TAD Pilha que armazene um **caractere** ou um **número real** (Pilha **HETEROGÊNEA**) usando alocação **dinâmica/encadeada** com as mesmas operações do exercício anterior.

Utilizando esse TAD, desenvolver um programa aplicativo para manipulação de expressões matemáticas envolvendo: variáveis literais de A a F; operadores de adição (+), subtração (-), divisão (/), multiplicação (*) e potenciação (^); e os delimitadores de escopo parênteses, colchetes e chaves. O programa inicia com a entrada da expressão, na forma infixa, pelo usuário. Em seguida, o programa deve realizar as seguintes operações:

- **Validação de escopo:** percorrer a expressão verificando se os escopos estão sendo abertos e fechados corretamente **considerando a precedência entre os delimitadores**, ou seja, se os escopos dos colchetes abrangem os parênteses e os escopos das chaves abrangem os colchetes. Por exemplo:
  - Expressão **[({A+D)/B}*F]** não é válida (ordem dos fechamentos divergem da ordem das aberturas).
  - Expressão **[({A+D}/B)*F]** não é válida (precedência não é obedecida)

- Expressão **{[(A+D)/B]*F}** é válida

Ao final do processo, o programa deve emitir uma mensagem com o resultado da validação e, no caso de erro, indicar qual o problema encontrado.

- **Conversão da expressão:** realizar a conversão da expressão para a forma pós-fixa. A expressão resultante também deve ser mostrada na tela.

- **Avaliação da expressão:** o programa deve solicitar ao usuário os valores para as literais utilizadas na expressão, resolvê-la utilizando esses valores e mostrar o resultado obtido ou uma mensagem indicando algum erro encontrado (falta de operando ou de operador). Exemplos de falha:
  - **(A+D)*(^C)** não é válida (número de operandos não é adequado)
  - **(AD)/B^F** não é válida (número de operadores não é adequado)

3) Implemente um TAD Fila de carros usando alocação **estática/sequencial com o uso de um contador** e contemplando todas as operações básicas e a operação ***tamanho*** que retorna o tamanho da fila passada como entrada. Para cada carro devem ser guardados os seguintes dados: placa, tipo do serviço (A – avulso ou M – mensalista) e hora de entrada.

Utilizando esse TAD, desenvolver um programa aplicativo para controlar a entrada e saída de veículos em um estacionamento com as seguintes características:

- O estacionamento é composto por 5 boxes, sendo que em cada box pode ser estacionado até 10 veículos enfileirados.
- Para retirar um veículo do meio de um box, é necessário retirar os veículos que estão na frente e colocá-los novamente no final do box.

Além da opção de encerramento, a aplicação deverá apresentar um menu com as seguintes funcionalidades:

a) **Entrada de veículos**: onde o atendente (usuário) cadastrará os dados do veículo que está entrando no estacionamento. Esse veículo deve ser colocado no box mais vazio, visando minimizar o remanejamento de veículos na retirada. Quando os 5 boxes estiverem cheios, o veículo deve ser colocado em uma fila de espera para mais 10 carros. Acima desta capacidade, o estacionamento rejeita o veículo.

b) **Saída de veículos**: onde o atendente indica a placa do veículo que está saindo. A partir desta placa, o programa deve localizar o veículo e retirá-lo do estacionamento, fazendo a relocação necessária dos veículos dos boxes e da fila de espera (não pode ter carro na fila de espera se houver vaga em algum dos boxes). Se o carro for avulso (tipo de serviço), o sistema deve calcular o valor a ser pago. Atualmente, o estacionamento está cobrando R$ 10,00 para a primeira hora e R$ 3,00 para cada hora adicional, inclusive fração (com tolerância de 15 minutos). Ou seja, se passar 15 minutos da hora cheia, cobra-se uma nova hora.

c) **Visualização do cenário**: apresenta a situação do estacionamento, mostrando a disposição dos veículos estacionados em cada box e na fila de espera.

4) Implemente um TAD Fila de pessoas a serem vacinadas, usando alocação **dinâmica/encadeada (encadeamento simples),** contemplando as mesmas operações do exercício anterior. O registro de cada pessoa deve conter: o nome, a idade, se possui comorbidade (1 – sim ou 0 - não), o tipo da vacina reservada (A- Astrazeneca, P- Pfizer ou C- Coronavac), as datas previstas para a 1ª e 2ª doses e o local de vacinação (1- UTC1, 2- UTC2, 3- Sabiazinho ou 4- prefeitura). Além do TAD Fila, a aplicação também precisará de uma TAD Lista não ordenada, também do tipo pessoas, implementada usando a forma escolhida pelo grupo, desde que permita o cadastro de pelo menos 50 pessoas, e com todas as operações básicas e duas operações de remoção:

- ***menor_data()*** que recebe o endereço da lista como entrada e retorna os dados da pessoa com a menor data preenchida para a 2ª dose, removendo-a da lista. Em caso de empate, deve-se respeitar a ordem de cadastro na lista, ou seja, remover a primeira ocorrência. Caso não exista nenhum registro com essa data preenchida, a operação deve retornar falha.

- ***maior_idade()*** que recebe o endereço da lista e o local de vacinação como entrada e retorna os dados da pessoa com maior idade que irá ser vacinada naquele local, removendo-a da lista. Em caso de empate, deve-se respeitar a ordem de cadastro na lista, ou seja, remover a primeira ocorrência. Caso não exista nenhum registro para o local indicado, a operação deve retornar falha.

A partir desses TADs, desenvolva um programa para gerenciar a vacinação da COVID-19 em Uberlândia com as seguintes funcionalidades:

a) **Cadastro da lista de espera**: preenche os dados de uma pessoa na lista de espera da vacinação, criada no início da execução do programa. Durante esse cadastro, será solicitado ao usuário o nome e a idade do paciente, bem como se ele possui alguma comorbidade ou não. De acordo com essas informações, o sistema define o local de vacinação, conforme a regra abaixo:

   - Pessoas com comorbidades são vacinadas no UTC – entrada pela Av. Getúlio Vargas (UTC1).
   - Pessoas sem comorbidades com até 60 anos são vacinadas no UTC – entrada pela Av. Cipriano del Fávero (UTC2).
   - Pessoas sem comorbidades acima de 60 anos são vacinadas na prefeitura.

   A Arena Sabiazinho será destinada apenas para a aplicação das 2ª doses. Os demais dados (datas e tipo da vacina) são preenchidos ao longo do processo.

b) **Geração das filas diárias:** determina as pessoas que serão vacinadas no dia seguinte. Seu funcionamento consiste em selecionar da lista de espera as 6 pessoas mais velhas para cada local de vacinação da 1ª dose e 12 para o Sabiazinho. Essas pessoas devem ser retiradas da lista e colocadas na fila do respectivo local, ou seja, cada ponto de vacinação terá a sua fila (criadas previamente). Nesse processo, também será definida, de forma aleatória, qual

vacina será aplicada, sendo que são disponibilizadas duas doses de cada vacina por local, exceto pelo Sabiazinho que recebe 4 doses diárias de cada vacina.

c) **Controle de vacinação**: corresponde ao processo de vacinação de uma pessoa. Ao ser selecionada, essa opção alternará entre os locais de vacinação, de forma iterativa, até que todas a filas estejam vazias (finalizar a vacinação do dia).

- Se o local corrente (selecionado) for destinado à aplicação da 1ª dose, o sistema deve retirar a primeira pessoa da fila correspondente, preencher a data da 1ª dose com a data corrente e calcular a data para 2ª dose conforme a tabela abaixo:

Vacina	Astrazeneca	Pfizer	Coronavac	Janssen	Sputnik-V
Prazo 2ª dose	90 dias	Não tem	60 dias	30 dias	45 dias

Em seguida, mudar o local de vacinação para Sabiazinho (onde são aplicadas a 2ª dose) e inserir novamente o registro na lista de espera.

- Se o local selecionado for o Sabiazinho, o sistema retira a primeira pessoa da fila e a inclui em uma lista de vacinados, a qual também é inicializada no início do programa.

d) **Cenário da vacinação**: essa opção do menu é responsável por apresentar na tela a situação atual das listas e filas. No caso das filas, deve mostrar o local de vacinação e, em seguida, os dados das pessoas e sua posição na fila. No caso das listas, indicar o nome da lista e, em seguida, os dados das pessoas na ordem com a qual aparecem na lista.

5) Implemente um Fila de Prioridade Ascendente (FPA) de alunos, usando alocação **dinâmica/encadeada (encadeamento simples) e remoção ordenada**. A estrutura aluno deve ser disponibilizada no arquivo cabeçalho do TAD para facilitar as operações e é formada pelos campos: matrícula, nome, CRA e ano de ingresso (campo usado na ordenação). O TAD deve contemplar as operações: cria_fp, fp_vazia, fp_cheia, insere, remove e esvazia_fp. Utilizando as operações disponibilizadas no TAD, implemente um programa aplicativo que permita a criação de uma FPA, a inserção e remoção de alunos, bem como esvaziar e imprimir a FPA. Essas ações devem ser executadas repetidamente até que o usuário solicite a saída do sistema, exceto pela criação da FPA que só pode ser executada uma única vez e antes de qualquer outra.

6) Implemente uma Fila de Prioridade Descendente (FPD) de transplantes, usando alocação **estática/sequencial (uso do contador) e inserção ordenada**. A FPD deve conter no máximo 20 pacientes, cada qual com as seguintes informações: nome, idade, órgão desejado (1 - coração, 2 - fígado, 3 – rins ou 4 – córnea) e grau de

gravidade da doença (campo usado na ordenação). Devem ser implementadas as mesmas operações do TAD (quando aplicável) e opções de menu usadas no exercício anterior.

7) Implementar o TAD Deque de números inteiros usando alocação **estática/sequencial (com desperdício de posição)** com no máximo 15 elementos. O TAD deve contemplar as operações: cria_deque, deque_vazio, deque_cheio, insere_inicio, insere_final, remove_inicio, remove_final e apagar_deque (que libera a memória alocada para o deque). Além disso, deve-se implementar um programa aplicativo que permita o usuário realizar repetidamente as seguintes ações: criar um deque, inserir e remover elementos, imprimir o conteúdo do deque e liberá-lo. A opção de criação só pode ser executada se o deque não existe (início do programa ou após usa liberação). As demais opções só podem ser executadas após a criação de um deque. O programa também deve ter uma opção para encerrar a execução.

8) Implementar a TAD Deque de caracteres usando alocação **dinâmica/encadeada (encadeamento circular).** O TAD deve contemplar as mesmas operações e programa aplicativo do exercício anterior.