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 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) Implementar a TAD Pilha de **números inteiros** usando alocação **estática/sequencial** com todas as operações básicas (cria_pilha, pilha_vazia, pilha_cheia, push, pop, ve_topo) e a operação ***esvazia_pilha***: que recebe o endereço de uma pilha e elimina seus elementos, retornando-a 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) Implementar a TAD Pilha de **caracteres** usando alocação **dinâmica/encadeada** com as operações básicas e a ***esvazia_pilha***: que recebe o endereço de uma pilha e elimina seus elementos, retornando-a para o estado de vazia. Utilizando esse TAD, implementar um programa aplicativo para manipulação de expressões matemáticas envolvendo: variáveis literais de A a J; 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}*J]** não é válida (ordem dos fechamentos divergem da ordem das aberturas).
  - Expressão **[({A+D}/B)*J]** não é válida (precedência não é obedecida)
  - Expressão **{[(A+D)/B]*J}** 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 e imprimir a expressão resultante.

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

3) Implemente um TAD Fila usando alocação **estática/sequencial com a estratégia com uso de um contador** e contemplando todas as operações básicas e a operação ***tamanho*** (que retorna o tamanho da fila passada como entrada). A especificação da estrutura de representação implementada deve atender aos requisitos necessários ao desenvolvimento de um programa para controlar a entrada e saída de veículos em um estacionamento. Para isso, considere as seguintes informações:

- 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, o programa deverá apresentar um menu com as seguintes funcionalidades:

a) **Entrada de veículos**: onde o usuário cadastrará a placa 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 o estacionamento estiver cheio, 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 usuário 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 boxes com vaga).

c) **Visualização do cenário**: apresenta a situação atual do estacionamento, mostrando os veículos estacionados e suas respectivas vagas/box, incluindo a fila de espera.

4) Implementar a TAD Fila usando alocação **dinâmica/encadeada simples,** contemplando as mesmas operações e o programa aplicativo do exercício 5. Utilizando esse TAD, desenvolva um programa para controlar a abertura e o fechamento dos caixas de acordo com o tempo de espera em uma fila única de supermercado. Este programa deverá possuir as seguintes funcionalidades:

a) **Simulador de fila**: responsável por simular a chegada de clientes na fila única. Ao ser selecionada no menu, essa opção solicita ao usuário a quantidade de clientes a serem inseridos na fila. O registro de cada cliente é formado pelo tempo que o caixa levará para atendê-lo, o tempo de espera na fila e o horário de sua entrada na fila (horário do sistema). O tempo de atendimento é definido aleatoriamente, variando de 2 a 10 unidades de tempo (unidade de simulação).

b) **Simulador de tempo:** responsável por fazer o controle dos caixas e da fila. Cada vez que for selecionada no menu, essa opção deve simular o avanço de uma unidade de tempo. Isso envolve aumentar o tempo de atendimento de cada caixa aberto e o tempo de espera de cada cliente na fila. Após o término do tempo previsto para o atendimento do cliente atualmente no caixa, ele se torna disponível. O atendimento ao próximo cliente da fila ocorrerá quando houver algum caixa aberto e disponível.

c) **Otimizador de caixas**: responsável pelo controle da abertura e do fechamento de até 8 caixas, visando manter o menor número de caixas abertos sem comprometer a "satisfação" do cliente. Após uma pesquisa, descobriu-se que o cliente fica satisfeito quando o tempo de espera na fila não ultrapassa 30 unidades de tempo e a fila não têm mais que 15 pessoas. Portanto, um novo caixa é aberto quando a fila tem 80% dessa capacidade de pessoas e o tempo médio de espera dos últimos 5 clientes é de 90% do limite especificado. Por outro lado, um caixa é fechado quando ambas as métricas está abaixo de 60%. Esse processo é automático (não tem opção no menu) e deve ser realizado ao final de cada simulação de tempo.

d) **Visualizador de cenário**: essa opção do menu é responsável por apresentar na tela a situação atual da fila e dos caixas. Para isto, as seguintes informações são necessárias: quantidade de caixas abertos, quantidade de pessoas na fila, o tempo médio de atendimento dos últimos 5 clientes.

5) Usando alocação **estática/sequencial (estratégia com desperdício de posição) e remoção ordenada**, implemente um Fila de Prioridade Ascendente (FPA) com no máximo 20 alunos. A estrutura aluno também deve estar encapsulada no TAD e é formada pelos campos: matrícula, nome, CRA, carga horária semanal e ano de ingresso. O TAD deve contemplar as operações: cria_fp, fp_vazia, fp_cheia, insere, remove (ordenada pelo ano de ano de ingresso) e esvazia_fp. Rambém deve ser implementado um programa aplicativo que, utilizando as operações disponibilizadas no TAD, 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 deve ocorrer uma vez e antes de qualquer outra.

6) Usando alocação **dinâmica/encadeada (encadeamento circular) e inserção ordenada**, implemente uma Fila de Prioridade Descendente (FPD) de pacientes. A estrutura criada deve conter o nome e a idade do paciente, o órgão necessário (coração, fígado, rins, córnea ou pulmão), o grau de gravidade da doença. O TAD deve considerar as mesmas operações do exercício anterior. Com base na FPD,

desenvolva um programa para controlar o fornecimento de órgãos para transplante. Este programa deverá gerenciar o cadastro de pacientes, os quais serão organizados em filas de acordo com o tipo de órgão a ser transplantado (coração, fígado, rins, córnea e pulmão) e ordenados pela gravidade do quadro clínico e pela ordem de inclusão na fila. Além disso, o programa deve ter um sistema de cadastro de doadores, onde é inserido o nome do doador, os órgãos doados e, para cada órgão, qual o paciente beneficiado. Também deve ter a opção para apresentar a lista de doadores, por ordem de cadastro, e seus respectivos dados. Caso não exista pacientes necessitando de um determinado órgão, o mesmo deve entrar em uma fila de disponibilidade e, logo que apareça um paciente, o mesmo deve ser transplantado.

7) Implementar o TAD Deque de números inteiros usando alocação **estática/sequencial (uso do contador).** O TAD deve contemplar as operações: cria_deque, deque_vazio, deque_cheio, insere_inicio, insere_final, remove_inicio, remove_final e libera_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 números reais (*double*) usando alocação **dinâmica/encadeada (encadeamento simples).** O TAD deve contemplar as mesmas operações e programa aplicativo do exercício anterior.