Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Pato Branco

Professora: Rúbia Eliza de Oliveira Schultz Ascari
Departamento Acadêmico de Informática (Dainf)
Tecnologia em Análise e Desenvolvimento de Sistemas
Estruturas de Dados 2

Exercícios sobre Filas e Deques Encadeadas (Resolução)

1. Escreva uma função que recebe uma fila encadeada que guarda n números inteiros e verifica se a fila está ordenada (ordem crescente).

RESOLUÇÃO

itemFilaEnc.c

```
int filaOrdenada(Fila *f) {
    Celula *aux = f->ini;
    int ord = 1;
    while (aux != NULL) {
        if ((aux->prox != NULL) && (aux->item.chave > aux->prox-
>item.chave)) {
            ord = 0;
            break;
        }
        aux = aux->prox;
    }
    return ord;
}
```

usaTADFilaEnc.c

```
int ord = filaOrdenada(f);
    printf("\nFila esta ordenada? %d", ord);
```

2. Utilize uma estrutura TAD de fila encadeada para representar um sistema para controlar os chamados de suporte recebidos por uma empresa de Informática.
   a) Considere que a struct a ser criada para armazenar os dados deve ser nomeada com o SEUNOME e deve ter pelo menos três campos referentes a dados dos chamados, sendo um deles o nome do usuário.
   b) Crie um programa que simula a criação, inclusão, exclusão e impressão dos dados dos chamados, sendo cadastrados pelo menos quatro registros de contatos.

RESOLUÇÃO

Pode variar de acordo com os campos utilizados, mas essencialmente devem ser feito ajustes nas funções de inclusão (incluindo novos parâmetros de acordo com os campos criados) e impressão de dados (imprimindo os dados referentes aos novos campos).

3. Considere que você recebeu a incumbência de desenvolver um jornal digital (*newsletter*) interno com notícias da empresa em que trabalha. Novidades,

lançamentos, e conteúdos diversos de interesse dos funcionários podem ser indicados para inclusão nesse jornal. Assim, fazendo uso dos conteúdos estudados até o momento sobre estruturas de dados encadeadas, escolha uma estrutura que permita a inclusão de notícias no jornal, por ordem de recebimento, e também inclusão prioritária antes das existentes. Após armazenamento das notícias desejadas na estrutura de dados escolhida, deve ser possível ao usuário realizar a remoção do primeiro e do último elemento armazenado. Para isso faça:

a) Identifique a estrutura de dados encadeada mais adequada para atender ao que foi solicitado.
b) Crie um TAD com funções básicas (criação, inclusão, remoção, impressão, destruição) que permitam a manutenção de notícias para o jornal, armazenando na *struct* correspondente pelo menos quatro campos que julguem importantes, de tipos de dados numérico e textual. É obrigatório a inclusão de um campo que permita identificar a classificação da notícia (Ex.: Esporte, Política, Cultura, entre outros).
c) Apresente ao usuário um menu para interação com o TAD desenvolvido, permitindo a inclusão de notícias, impressão e remoção (início e final). d) Crie uma função para calcular a quantidade de notícias cadastradas por classificação, e o percentual correspondente à classificação com maior número de notícias.

RESOLUÇÃO

bib_newsletter.h

```
typedef struct item Item;
typedef struct celula Celula;
typedef struct deque Deque;
Deque *criaDeque();
int verificaDequeVazia(Deque* dq);
void insereFinalDeque(Deque* dq, int chave, int classificacao, float custo,
char titulo[200]);
void insereInicioDeque(Deque* dq, int chave, int classificacao, float custo,
char titulo[200]);
void removeInicio_Deque(Deque* dq);
int removeFinal_Deque(Deque* dq);
void imprime_Deque(Deque* dq);
void libera_Deque(Deque* dq);
```

bib_newsletter.c

```
#include <stdio.h>
#include <stdlib.h>
#include "bib_newsletter.h"

struct item
{
    int chave;
    int classificacao;
    float custo;
    char titulo[200];
};

struct celula
{
    Item item;
    Celula *prox;
};
```

```
struct deque
{
    Celula *ini;
    Celula *fim;
};

//Cria uma estrutura deque
Deque *criaDeque()
{
    Deque *dq = (Deque*) malloc(sizeof(Deque));
    if(dq != NULL)
    {
        dq->ini = NULL;
        dq->fim = NULL;
    }
    return dq;
}

//Verifica deque vazio
int verificaDequeVazia(Deque* dq)
{
    return (dq->ini == NULL);
}

void insereFinalDeque(Deque* dq, int chave, int classificacao, float custo,
char titulo[200])
{
    //cria novo item que vai ser guardado na Deque
    Item novo;
    novo.chave = chave;
    novo.classificacao = classificacao;
    novo.custo = custo;
    strcpy(novo.titulo, titulo);
    //cria nova célula que vai guardar o item
    Celula *nova = malloc(sizeof(Celula));
    nova->item = novo;
    if(verificaDequeVazia(dq))
    {
        nova->prox = dq->ini;
        dq->ini = nova;
    }
    else
    {
        nova->prox = NULL;
        dq->fim->prox = nova;
    }
    dq->fim = nova;
}

void insereInicioDeque(Deque* dq, int chave, int classificacao, float custo,
char titulo[200])
{
    //Cria novo item
    Item novo;
    novo.chave = chave;
    novo.classificacao = classificacao;
    novo.custo = custo;
    strcpy(novo.titulo, titulo);
    //Cria nova celula que vai ser guardada o item
    Celula *nova = malloc(sizeof(Celula));
    nova->item = novo;
    nova->prox = dq->ini;
     if (verificaDequeVazia(dq)) //se está vazia fim será igual a ini
```

```
    {
        dq->fim = nova;
    }
    dq->ini = nova;
}

void removeInicio_Deque(Deque* dq)
{
    if (verificaDequeVazia(dq))
    {
        printf("Erro: Deque vazia!\n");
        return;
    }

    Celula *remover = dq->ini;
    dq->ini = remover->prox;
    free(remover);
    if (verificaDequeVazia(dq)) //se ficou vazia, fim aponta para NULL
        dq->fim = NULL;
}

int removeFinal_Deque(Deque* dq)
{
    if (verificaDequeVazia(dq))
    {
        printf("Erro: Deque vazia!\n");
        return;
    }
    Celula *remover = dq->fim;

    if (remover == dq->ini) //remover o primeiro e unico elemento?
    {
        dq->ini = NULL;
        dq->fim = NULL;
    }
    else
    {
        Celula *ultimo = dq->ini;
        while (ultimo->prox != dq->fim)
        {
            ultimo = ultimo->prox;
        }

        ultimo->prox = NULL;
        dq->fim = ultimo;
    }
    free(remover);
}

void imprime_Deque(Deque* dq)
{
    Celula *aux = dq->ini;
    while (aux != NULL)
    {
        printf("Chave: %d", aux->item.chave);
        printf("\nTipo de Noticia: %d", aux->item.classificacao);
        printf("\nTitulo da noticia: %s\n", aux->item.titulo);
        printf("\nCusto: %f", aux->item.custo);
        aux = aux->prox;
    }
}

void libera_Deque(Deque* dq)
{
```

```
    Celula *aux = dq->ini;
    Celula *liberar;
    while(aux != NULL)
    {
        liberar = aux;
        aux = aux->prox;
        free(liberar);
    }
    free(dq);
}


void cadNoticia(Deque *dq)
{
    Celula *aux = dq->ini;
    int esportes = 0;
    int acidentes = 0;
    int moda = 0;
    int fofoca = 0;
    int total = 0;

    while (aux != NULL)
    {
        if (aux->item.classificacao == 1) {
            esportes++;
        }
        else if (aux->item.classificacao == 2) {
            acidentes++;
        }
        else if (aux->item.classificacao == 3) {
            moda++;
        }
        else if (aux->item.classificacao == 4) {
            fofoca++;
        }
        total++;
        aux = aux->prox;
    }
    //Calcular o percentual com o maior numero de noticias
    int var = 0;
    char tipo[50];
    if(esportes > var) {
        var = esportes;
        strcpy(tipo, "esportes");
    }
    if (var < acidentes) {
        var = acidentes;
        strcpy(tipo, "acidentes");
    }
    if (var < moda) {
        var = moda;
        strcpy(tipo, "moda");
    }
    if (var < fofoca){
        var = fofoca;
        strcpy(tipo, "fofoca");
    }

    int maior = ((var * 100) / total);
    printf("A noticia com maior porcentagem eh %s com %d porcento.", tipo,
maior);

}
```

main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "bib_newsletter.h"

int main()
{
    Deque *dq;
    int acao, chave, vazia;
    float custo;
    char titulo[200];
    int tipo;

    printf("Criando deque vazia.\n");
    dq = criaDeque();

    vazia = verificaDequeVazia(dq);
    printf("Deque vazia? %d\n", vazia);

    do
    {
        printf("Informe o que voce deseja fazer: \n1 - Inserir elementos no
inicio da lista.\n2 - Inserir elementos no final da lista.\n3 - Remover
elementos do comeco da lista.\n4 - Remover elementos do final da lista.\n5
- Imprimir a lista.\n6 - sair.\n:");
        scanf("%d", &acao);

        switch (acao)
        {
        case 1:
            printf("Informe a chave na noticia: ");
            scanf("%d", &chave);
            printf("Informe o custo da noticia: ");
            scanf("%f", &custo);
            printf("Informe o tipo da noticia:\n1 - Esportes. \n2 - Acidentes
\n3 - Moda \n4 - Fofocas ");
            scanf("%d", &tipo);
            setbuf(stdin, NULL);
            printf("Informe o titulo da noticia: ");
            gets(titulo);

            insereInicioDeque(dq, chave, tipo, custo, titulo);
            break;
        case 2:
            printf("Informe a chave na noticia: ");
            scanf("%d", &chave);
            printf("Informe o custo da noticia: ");
            scanf("%f", &custo);
            printf("Informe o tipo da noticia:\n1 - Esportes. \n2 - Acidentes
\n3 - Moda \n4 - Fofocas ");
            scanf("%d", &tipo);
            setbuf(stdin, NULL);
            printf("Informe o titulo da noticia: ");
            gets(titulo);

            insereFinalDeque(dq, chave, tipo, custo, titulo);
            break;
        case 3:
            removeInicio_Deque(dq);
```

```
            break;
        case 4:
            removeFinal_Deque(dq);
            break;
        case 5:
            imprime_Deque(dq);
            break;
        }
    }
    while(acao != 6);

    cadNoticia(dq);

    libera_Deque(dq);
    printf("\nLista liberada.");

    return 0;
}
```

4. Você está trabalhando em uma grande companhia que atua na área de recursos naturais. Usando imagens de satélite, a companhia deseja fazer um inventário de todas as árvores em uma determinada região, utilizando um deque para armazenar os dados das árvores. Para isso faça:
   a) Use como entrada para o problema o número de árvores na região mapeada e o código da espécie de cada árvore detectada.
   b) Calcule e apresente a porcentagem em que cada espécie de árvore aparece na região.
   c) Mostre qual é o código da espécie mais frequente.
   d) Ao percorrer os códigos de entrada, quando encontrar o código de uma espécie que já está armazenada na estrutura, em vez de armazená-la novamente, incremente um campo de quantidade no item.
   e) Considere que o programa deve permitir a remoção do último cadastro inserido (opção de desfazer), e nesse caso, precisa verificar se é necessário atualizar as quantidades já armazenadas de uma determinada espécie na mesma região.

Exemplo:

```
Informe a regiao que esta sendo inventariada: Pato Branco
Informe a especie de arvore que ira compor o inventario: Acer Palmatum
Informe a quantidade de arvores desta especie que foram identificadas: 5

Deseja armazenar mais dados para essa regiao? (S/N)S

Informe a especie de arvore que ira compor o inventario: Amoreira
Informe a quantidade de arvores desta especie que foram identificadas: 8

Deseja armazenar mais dados para essa regiao? (S/N)S

Informe a especie de arvore que ira compor o inventario: Ipe Amarelo
Informe a quantidade de arvores desta especie que foram identificadas: 9

Deseja armazenar mais dados para essa regiao? (S/N)N
Deseja armazenar dados de outra regiao? (S/N)S

Informe a regiao que esta sendo inventariada: Vitorino
Informe a especie de arvore que ira compor o inventario: Acer Palmatum
Informe a quantidade de arvores desta especie que foram identificadas: 6

Deseja armazenar mais dados para essa regiao? (S/N)S
```

```
Informe a especie de arvore que ira compor o inventario: Ipe Amarelo
Informe a quantidade de arvores desta especie que foram identificadas: 10

Deseja armazenar mais dados para essa regiao? (S/N)N
Deseja armazenar dados de outra regiao? (S/N)S

Informe a regiao que esta sendo inventariada: Pato Branco
Informe a especie de arvore que ira compor o inventario: Acer Palmatum
Informe a quantidade de arvores desta especie que foram identificadas: 3
* Identificado registro existente, atualizada a quantidade.

Deseja armazenar mais dados para essa regiao? (S/N)S

Informe a especie de arvore que ira compor o inventario: Pinheiro Negro
Informe a quantidade de arvores desta especie que foram identificadas: 8

Deseja armazenar mais dados para essa regiao? (S/N)N

Deseja armazenar dados de outra regiao? (S/N)N

===Especies de arvores identificadas por regiao inventariada ===
* Pato Branco
Chave: 1 - Especie Acer Palmatum - Quantidade: 8
* Pato Branco
Chave: 2 - Especie Amoreira - Quantidade: 8
* Pato Branco
Chave: 3 - Especie Ipe Amarelo - Quantidade: 9
* Vitorino
Chave: 4 - Especie Acer Palmatum - Quantidade: 6
* Vitorino
Chave: 5 - Especie Ipe Amarelo - Quantidade: 10
* Pato Branco
Chave: 7 - Especie Pinheiro Negro - Quantidade: 8

===Porcentagem de arvores por regiao===
Pato Branco
* 8 arvores da especie Acer Palmatum, equivalente a 24.24 % de arvores da
regiao (Total: 33).
Pato Branco
* 8 arvores da especie Amoreira, equivalente a 24.24 % de arvores da regiao
(Total: 33).
Pato Branco
* 9 arvores da especie Ipe Amarelo, equivalente a 27.27 % de arvores da
regiao (Total: 33).
Vitorino
* 6 arvores da especie Acer Palmatum, equivalente a 37.50 % de arvores da
regiao (Total: 16).
Vitorino
* 10 arvores da especie Ipe Amarelo, equivalente a 62.50 % de arvores da
regiao (Total: 16).
Pato Branco

* 8 arvores da especie Pinheiro Negro, equivalente a 24.24 % de arvores da
regiao (Total: 33).
```

RESOLUÇÃO

itemDequeEnc.c

```
int existe(Deque *dq, char regiao[], char especie[])
{
```

```
    Celula *auxBusca = dq->ini;
    while (auxBusca != NULL)
    {
        if ((strcmp(regiao, auxBusca->item.regiao) == 0) &&
                (strcmp(especie, auxBusca->item.especie) == 0))
        {
            return 1;
        }
        auxBusca = auxBusca->prox;
    }
    return 0;
}

void atualizaExistente(Deque *dq, char regiao[], char especie[], int novaQtd)
{
    Celula *auxBusca = dq->ini;
    while (auxBusca != NULL)
    {
        if ((strcmp(regiao, auxBusca->item.regiao) == 0) &&
                (strcmp(especie, auxBusca->item.especie) == 0))
        {
            printf("*  Identificado  registro  existente,  atualizada  a
quantidade.\n\n");
            auxBusca->item.qtd = auxBusca->item.qtd + novaQtd;
            return;
        }
        auxBusca = auxBusca->prox;
    }
}

void insereFinal_Deque(Deque* dq, int chave, char regiao[], char especie[],
int qtd)
{
    if (!existe(dq, regiao, especie))
    {
        //cria novo item que vai ser guardado na Deque
        Item novo;
        novo.chave = chave;
        strcpy(novo.regiao, regiao);
        strcpy(novo.especie, especie);
        novo.qtd = qtd;
        //cria nova célula que vai guardar o item
        Celula *nova = malloc(sizeof(Celula));
        nova->item = novo;
        if (verificaDequeVazia(dq)) //se está vazia adiciona no início
        {
            nova->prox = dq->ini;
            dq->ini = nova;
        }
        else
        {
            //senão aciciona no final
            //inserção no final, a nova célula aponta para NULL
            nova->prox = NULL;
            dq->fim->prox = nova;
        }
        dq->fim = nova;
    }
    else
    {
        atualizaExistente(dq, regiao, especie, qtd);
    }
}
```

```
void insereInicio_Deque(Deque* dq, int chave, char regiao[], char especie[],
int qtd)
{
    if (!existe(dq, regiao, especie))
    {
        //cria novo item que vai ser guardado na Deque
        Item novo;
        novo.chave = chave;
        strcpy(novo.regiao, regiao);
        strcpy(novo.especie, especie);
        novo.qtd = qtd;

        //cria nova célula que vai guardar o item
        Celula *nova = malloc(sizeof(Celula));
        nova->item = novo;
        nova->prox = dq->ini;
        if (verificaDequeVazia(dq)) //se está vazia fim será igual a ini
        {
            dq->fim = nova;
        }
        dq->ini = nova;
    }
    else
    {
        atualizaExistente(dq, regiao, especie, qtd);
    }

}

void imprime_Deque(Deque* dq)
{
    Celula *aux = dq->ini;
    while (aux != NULL)
    {
        printf("* %s\n", aux->item.regiao);
        printf("Chave: %d - Especie %s - Quantidade: %d \n", aux->item.chave,
aux->item.especie, aux->item.qtd);
        aux = aux->prox;
    }
}

void calculaPorcentagemEspecie(Deque *dq)
{
    int i = 0, total;
    float porcent = 0;
    //Contabiliza total de árvores registradas em uma mesma região
    Celula *aux = dq->ini;

    char regiaoAnt[30];
    strcpy(regiaoAnt, "");
    while (aux != NULL)
    {
        if (strcmp(aux->item.regiao, regiaoAnt) != 0)
        {
            Celula *auxBusca = dq->ini;
            total = 0;
            while (auxBusca != NULL)
            {
                if (strcmp(aux->item.regiao, auxBusca->item.regiao) == 0)
                {
                    total = total + auxBusca->item.qtd;
                }

                auxBusca = auxBusca->prox;
```

```
            }
        }
        strcpy(regiaoAnt, aux->item.regiao);
        printf("%s\n", aux->item.regiao);
        porcent = (aux->item.qtd * 100.0) / (float) total;
        printf("* %d arvores da especie %s, equivalente a %.2f %% de arvores
da regiao (Total: %d).\n",
               aux->item.qtd, aux->item.especie, porcent, total);
        aux = aux->prox;
    }
}
```

usaTADDequeEnc.c

```
#include <stdio.h>
#include <stdlib.h>
#include "itemDequeEnc.h"

int main()
{

    Deque* dq = cria_Deque();

    char opcao, opRegiao;
    char regiao[30], especie[30];
    int nArvores, chave = 0;

    do
    {
        printf("Informe a regiao que esta sendo inventariada: ");
        gets(regiao);
        do
        {
            printf("Informe a especie de arvore que ira compor o inventario:
");
            gets(especie);
            printf("Informe a quantidade de arvores desta especie que foram
identificadas: ");
            scanf("%d", &nArvores);
            chave++;
            insereFinal_Deque(dq, chave, regiao, especie, nArvores);
            setbuf(stdin, NULL);

            printf("Deseja armazenar mais dados para essa regiao? (S/N)");
            scanf("%c", &opcao);
            setbuf(stdin, NULL);

        }
        while (opcao == 'S');

        printf("Deseja armazenar dados de outra regiao? (S/N)");
        scanf("%c", &opRegiao);
        setbuf(stdin, NULL);
    }
    while (opRegiao == 'S');

    printf("===Especies de arvores identificadas por regiao inventariada
===\n");
    imprime_Deque(dq);
    printf("\n===Porcentagem de arvores por regiao===\n");
    calculaPorcentagemEspecie(dq);

    libera_Deque(dq);
```

```
    return 0;
}
```