UERJ-ZO
Disciplina: Estrutura de Dados I
Professor: Raul Carlos Costa

Nome do Aluno: Jonatha Salles Menezes / Matrícula: 2211312125

AV2 – Estrutura de Dados

Fila estática

Pilha estática

Pilha dinâmica

Rio de Janeiro / RJ
Fevereiro, 2023

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <malloc.h>
#define TAM_VET 10
#define VERD 1
#define FALSO 0
#define ERRO -1

int main();

typedef struct Item
{
   int numero;
   struct Item *anterior;
} Elemento;

struct Dados
{
   int dado[TAM_VET];
   int tamanho;
   int topo;
   int inicio;
};

int Menu()
{
   int escolha;
   do
   {
      printf("==================");
      printf("\nEscolha uma opcao: \n");
      printf("[1]Fila \n");
      printf("[2]Pilha \n");
      printf("[3]Pilha Dinamica\n[4]Encerrar\n--> ");
      scanf("%d", &escolha);
   }while (escolha < 1 || escolha > 4);
   return escolha;
}

void LimpaDados(struct Dados* D)
{
   int i;
   for(i=0; i<TAM_VET; i++)
   {
      D->dado[i] = 0;
   }
   D->tamanho = 0;
```

```
    D->inicio = 0;
    D->topo = -1;
}

int FilaVazia(struct Dados *D)
{
    if(D->tamanho > 0) return FALSO;
    return VERD;
}

int FilaCheia(struct Dados *D)
{
    if (D->tamanho < 10) return FALSO;
    return VERD;
}

void InserirFila(struct Dados *D)
{
    printf("Digite o elemento a ser inserido na FILA \n--> ");
    D->topo++;
    scanf("%i", &D->dado[D->topo]);
    D->tamanho++;
}

void RemoverFila(struct Dados *D)
{
    printf("Elemento removido e: %i\n", D->dado[D->inicio]);
    D->inicio++;
    D->tamanho--;
}

void MostrarFila(struct Dados *D)
{
    int i;
    printf("\nFila: [");
    for(i = D->inicio; i <= D->topo; i++)
    {
        printf("%i", D->dado[i]);
        if(i != D->topo)printf(", ");
    }
    printf("]\n");
}

int SubmenuFila()
{
    int opcao;
    do
    {
```

```
        printf("==================\n");
        printf("O que deseja fazer?\n[1]Adicionar\n[2]Remover\n[3]Mostrar\n[4]Reiniciar\n[5]Fila
esta cheia?\n[6]Fila esta vazia?\n[7]Voltar\n--> ");
        scanf("%i", &opcao);
    }while (opcao < 0 || opcao > 7);
    return opcao;
}

void FilaFunc ()
{
    int opcao;
    int ok;
    struct Dados D;
    LimpaDados(&D);
    while(1)
    {
        opcao = SubmenuFila();
        if(opcao == 7) break;
        if(opcao == 1)
        {
            ok = FilaCheia(&D);
            if(ok == VERD) printf(" Fila Cheia!!!\n ");
            else
            {
                InserirFila(&D);
                printf(" Elemento inserido com sucesso!!!\n ");
            }
        }
        else if(opcao == 2)
        {
            ok = FilaVazia(&D);
            if(ok == VERD) printf("Remocao invalida! Fila vazia\n");
            else RemoverFila(&D);
        }
        else if(opcao == 3)
        {
            MostrarFila(&D);
        }
        else if(opcao == 4)
        {
            LimpaDados(&D);
            printf("Fila reiniciada.\n");
        }
        else if(opcao == 5)
        {
            int estaCheia = FilaCheia(&D);
            if(estaCheia == VERD) printf("A filha esta cheia!\n");
            else printf("A fila nao esta cheia!\n");
```

```
        }
        else if(opcao == 6)
        {
            int estaVazia = FilaVazia(&D);
            if(estaVazia == VERD) printf("A fila esta vazia!\n");
            else printf("A fila nao esta vazia!\n");
        }
    }
    main();
}

void Empilha(int dado, struct Dados *D)
{
    if (D->tamanho == TAM_VET)
    {
        printf("Pilha cheia.\n");
        return;
    }
    else
    {
        D->topo++;
        D->dado[D->topo] = dado;
        D->tamanho++;
        return;
    }
}

int Desempilha(struct Dados *D)
{
    int aux;
    if (D->tamanho == 0)
    {
        printf("Pilha Vazia.\n");
        return;
    }
    else
    {
        aux = D->dado[D->topo];
        printf("\nElemento removido : %i \n", aux);
        D->topo--;
        D->tamanho--;
        return;
    }
}

void ImprimePilha(struct Dados *D)
{
    int i;
```

```
    printf("[");
    for(i=0; i <= D->topo; i++)
    {
        printf("%i", D->dado[i]);
        if (i != D->topo) printf(", ");
    }
    printf("]");
    printf("\n");
}

int PilhaCheiaOuVazia(int verificar, struct Dados *D)
{
    if(verificar == 0)
    {
        if (D->tamanho == TAM_VET) return 1;
        return 0;
    }
    else
    {
        if (D->tamanho == 0) return 1;
        return 0;
    }
}

void PilhaFunc()
{
    int op = -1;
    int dado, l;
    struct Dados D;
    LimpaDados(&D);
    while( op!= 7 )
    {
        printf("====================\n");
        printf( "O que deseja
fazer?\n[1]Adicionar\n[2]Remover\n[3]Mostrar\n[4]Reiniciar\n[5]Pilha esta cheia?\n[6]Pilha
esta vazia?\n[7]Voltar\n-->  ");
        scanf("%d", &op);
        if( op== 1)
        {
            printf( "\nDigite um numero: ");
            scanf("%i", &dado);
            Empilha(dado, &D);
        }
        else if( op== 2 )
        {
            Desempilha(&D);
        }
        else if( op== 3 )
```

```
        {
            ImprimePilha(&D);
        }
        else if(op ==  4)
        {
            LimpaDados(&D);
        }
        else if(op ==  5)
        {
            int resposta = PilhaCheiaOuVazia(0, &D);
            if(resposta == 1) printf("Pilha cheia!\n");
            else printf("A pilha nao esta cheia!\n");
        }
        else if(op == 6)
        {
            int resposta = PilhaCheiaOuVazia(1, &D);
            if(resposta == 1) printf("Pilha vazia!\n");
            else printf("A pilha nao esta vazia!\n");
        }
    }
    main();
}

void InicializarPilhaDinamica(Elemento **topo)
{
    *topo = NULL;

}

int PilhaDinamicaEstaVazia(Elemento **topo, struct Dados D)
{
    if(D.tamanho == 0)
        return VERD;
    else
        return FALSO;
}

int PegarElemento()
{
    int elemento;
    printf("Digite o elemento a ser adicionado\n-> ");
    scanf("%i", &elemento);
    return elemento;
}

void EmpilharPilhaDinamica(Elemento **topo, struct Dados *D)
{
    if(PilhaDinamicaEstaCheia(*D)) {printf("Pilha cheia! Impossivel empilhar mais\n"); return;}
```

```
    D->topo++;
    int novoNumero = PegarElemento();
    Elemento *novo;

    novo = (Elemento *) malloc(sizeof(Elemento));

    novo -> numero = novoNumero;
    D->dado[D->topo] = novoNumero;
    D->tamanho++;

    novo -> anterior = *topo;

    *topo = novo;
    printf("Elemento adicionado!\n");
}

int DesempilharPilhaDinamica(Elemento **topo, struct Dados *D)
{
    Elemento *antigo;
    antigo = *topo;
    int result;
    if(PilhaDinamicaEstaVazia(topo, *D))
    {
        printf("\n Pilha vazia! \n");
        return -1;
    }
    result = (*topo)->numero;
    *topo = (*topo)->anterior;
    free(antigo);
    D->tamanho--;
    D->topo--;
    return result;
}

void MostraPilhaDinamica(Elemento *topo, struct Dados *D)
{
    int i = 0;
    Elemento *item;

    printf("\n\n Listando...\n\n");
    printf("----------------------------------------\n");
    if (PilhaDinamicaEstaVazia(&topo, *D)){
        printf("A Pilha esta vazia!\n");
    }
    else {
        int i;
        item = topo;
        printf("\nItem     Valor    Endereco Ativo          Endereco Anterior\n");
```

```
        for(i=0; i <= D->topo; i++)
        {
            printf("[%2d] -> %2d        :%p                 :%p\n", i, D->dado[i], item, item->anterior);
            item = item->anterior;
        }
    }
    printf("-----------------------------------------\n\n");
}

int PilhaDinamicaEstaCheia(struct Dados D)
{
    if(D.tamanho == TAM_VET) return VERD;
    return FALSO;
}

void PilhaDinamicaFunc()
{
    int opcao;
    Elemento *topo;
    InicializarPilhaDinamica(&topo);
    struct Dados D;
    LimpaDados(&D);
    do
    {
        printf("O que deseja
fazer?\n[1]Adicionar\n[2]Remover\n[3]Mostrar\n[4]Reiniciar\n[5]Pilha dinamica esta
vazia?\n[6]Pilha dinamica esta cheia?\n[7]Voltar\n-->  ");
        scanf("%i", &opcao);
        if(opcao == 1)
        {
            EmpilharPilhaDinamica(&topo, &D);
        }
        else if(opcao == 2)
        {
            int elementoDesempilhado = DesempilharPilhaDinamica(&topo, &D);
            if (elementoDesempilhado != ERRO) printf("Elemento [%i] removido\n",
elementoDesempilhado);
            else printf("\nErro na remocao, pilha vazia");
        }
        else if(opcao == 3)
        {
            MostraPilhaDinamica(topo, &D);
        }
        else if(opcao == 4)
        {
            LimpaDados(&D);
            InicializarPilhaDinamica(&topo);
        }
```

```
        else if(opcao == 5)
        {
            int estaVazia = PilhaDinamicaEstaVazia(&topo, D);
            if(estaVazia == VERD) printf("A pilha dinamica esta vazia\n");
            else printf("A pilha dinamica nao esta vazia\n");
        }
        else if(opcao == 6)
        {
            int estaCheia = PilhaDinamicaEstaCheia(D);
            if(estaCheia == VERD) printf("A pilha dinamica esta cheia!\n");
            else printf("A pilha dinamica nao esta cheia\n");
        }
    }while (opcao != 7);
    main();
}

int main()
{
    setlocale(LC_ALL, "");
    int escolha;
    escolha = Menu();
    if (escolha == 1) FilaFunc();
    else if(escolha == 2) PilhaFunc();
    else if(escolha == 3) PilhaDinamicaFunc();
    //O programa só chega até aqui com a escolha 4
    printf("\n\tPrograma encerrado.\n\n");
    exit(0);
    return 0;
}
```