**2 - 1** Implemente uma pilha com vetor, conforme a interface dada *pilha.h*:

1. O TAD TPilha, com vetor de 100 elementos e a variável com a posição do topo;
2. Operação de criar;
3. Operação de empilhar;
4. Operação de desempilhar;
5. Operação que retorna tamanho;
6. Operação que retorna topo.

|  |
| --- |
| #ifndef PILHA\_H\_INCLUDED  #define PILHA\_H\_INCLUDED  typedef struct Pilha TPilha;  TPilha \*criar(int valor);  void empilhar(TPilha \*pilha, int valor);  int desempilhar(TPilha \*pilha);  int tamanho(TPilha \*pilha);  int topo(TPilha \*pilha);  #endif  #include <stdio.h>  #include <stdlib.h>  #include "pilha.h"  struct Pilha{  int \*vetor;  int quantidade, topo;  };  TPilha \*criar(int valor){  TPilha \*pilha = (TPilha \*) malloc(sizeof(TPilha));  pilha->vetor = malloc(sizeof(int)\*valor);  pilha->quantidade = valor;  pilha->topo = -1;  return pilha;  }  void empilhar(TPilha \*pilha, int valor){  if(pilha->topo < pilha->quantidade - 1){  pilha->topo = pilha->topo + 1;  pilha->vetor[pilha->topo] = valor;  }  }  int desempilhar(TPilha \*pilha){  if(pilha->topo > - 1){  pilha->topo = pilha->topo - 1;  return pilha->vetor[pilha->topo + 1];  }  }  int tamanho(TPilha \*pilha){  return pilha->quantidade;  }  int topo(TPilha \*pilha){  return pilha->vetor[pilha->topo];  } |

**2 - 2** Inclua na interface pilha.h e implemente a operação “esta\_vazia()”, a qual retorna 0 se a pilha não estiver vazia e 1 em caso contrário.

|  |
| --- |
| int pilhaVazia(TPilha \*pilha);  int estaVazia(TPilha \*pilha){  if(pilha->topo == -1){  return 0;  }  return 1;  } |

**2 - 3** Desenvolva uma função que receba uma TPilha como parâmetro e inverta seu conteúdo, utilizando uma pilha auxiliar.

|  |
| --- |
| void inverterPosicoes(TPilha \*pilha){  TPilha \*pilhaAuxiliar = criar(pilha->quantidade);  TPilha \*pilhaAuxiliar2 = criar(pilha->quantidade);  int i,j, index;  for(i = 0; i <= pilha->quantidade; i++){  empilhar(pilhaAuxiliar,desempilhar(pilha));  }  for(j = 0; j <= pilha->quantidade; j++){  empilhar(pilhaAuxiliar2,desempilhar(pilhaAuxiliar));  }  for(index = 0; index <= pilha->quantidade; index++){  empilhar(pilha,desempilhar(pilhaAuxiliar2));  }  } |

**2 - 4** Desenvolva uma função que receba uma TPilha e um valor n como parâmetros e remova a primeira ocorrência de n da pilha, se houver. A função poderá utilizar uma pilha auxiliar e só poderá utilizar as operações de pilha para manipular seus elementos.

|  |
| --- |
| int removerValor(TPilha \*pilha, int valor){  if(estaVazia(pilha) == 1){  TPilha \*pilhaAuxiliar = criar(pilha->quantidade);  int i,j;  for(i = 0; i <= pilha->quantidade; i++){  if(topo(pilha) == valor){  desempilhar(pilha);  }  empilhar(pilhaAuxiliar,desempilhar(pilha));  }  for(j = 0; j <= pilhaAuxiliar->quantidade; j++){  empilhar(pilha, desempilhar(pilhaAuxiliar));  }  }  } |