ESPECIFICACIÓN FORMAL TDA PILA DOBLE ESTÁTICA

Nombre:

TAD Pila Doble Estàtica (una estructura de un arreglo en el cual se encuentran 2 pilas estáticas, donde una de las pilas de N elementos crece de izquierda a derecha y la otra crece de derecha a izquierda.)

Sintaxis:

crearArreglo()->Arreglo

esLleno(Arreglo)--> boolean

esVacio(Arreglo)--> boolean

mostrarElementos(Arreglo)--> void

crearPila() -> Pila

esVacia(Pila)-> boolean

esLlena(Pila)->boolean

push(Pila, elemento)-> Pila

pop(Pila)->Pila

Semántica:

• Dados los elementos de una pila Pila donde el tope izquierdo tiene el valor -1 y el tope derecho tiene el valor de 12 se ingresan elementos e0. . . eN a la pila por la izquierda y por la derecha. Cada elemento corresponde al conjunto de datos del mismo tipo de dato entero.

crearPila() => inicializar Pila con un tope en el índice -1 y el otro tope con índice en 12.

esVacia(Pila)-> si la Pila no contiene ningún elemento, se retorna un True o un 1 indicando que está vacía, de lo contrario se retorna un false o 0.

esLlena(Pila)->si el tope de la izquierda es mayor o igual al tope de la derecha entonces la pila está llena y se indica a través de un print. De otra manera, se retorna un 0 que quiere decir que la Pila no está llena todavía.

push(Pila, elemento)-> if Pila no esLlena entonces aumentar el tope izquierdo y reducir el tope derecho, al índice se asigna el elemento ingresado, luego se retorna Pila.

pop(Pila)->if Pila no esVacia entonces disminuir en uno ambos topes.

TDA PILA

#define MAX\_PILA 11

typedef struct pila {

int topeizq;

int topeder;

int A[MAX\_PILA];

}Pila;

/\*PROTOTIPOS\*/

Pila crearPila();

int esVacia(Pila p);

int esLlena(Pila p);

Pila push(Pila p, int elem);

Pila pop(Pila p);

//FUNCIONES

Pila crearPila() {

Pila p;

p.topeizq=-1;

p.topeder=12;

return p;

}

int esVacia(Pila p) {

if (p.topeizq==-1) {

printf("\tPila vacia\n");

return 1;

} else {

return 0;

}

}

int esLlena(Pila p) {

if (p.topeizq>= p.topeder) {

printf("\nPila llena\n");

return 1;

} else {

return 0;

}

}

Pila push(Pila p, int elem) {

if (!esLlena(p)) {

p.topeizq++; //el tope es el indice que se va agregando

printf("Tope izq=%d\n",p.topeizq);

p.topeder--;

printf("Tope der=%d\n",p.topeder);

p.A[p.topeizq] = elem;

p.A[p.topeder] = elem;

}

return p;

}

Pila pop(Pila p) {

if (!esVacia(p)) {

p.topeizq--;

p.topeder--;

}

return p;

}

=================

/\*MAIN \*/

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include <math.h>

#include "pila.h"

#define MAX\_PILA 11

//PILA DOBLE ESTATICA

int main() {

Pila pila= crearPila(); //la var p de pila.c viene aqui

printf("\n Se ha creado la pila. Tope Izquierda= %d\t Tope Derecha= %d\n", pila.topeizq, pila.topeder);

printf("Esta vacia la pila?: \n");

while (esVacia(pila)) {

pila=push(pila,10);

pila=push(pila,20);

pila=push(pila,30);

pila=push(pila,40);

pila=push(pila,50);

pila=push(pila,60);

pila=push(pila,70);

pila=push(pila,80);

pila=push(pila,90);

pila=push(pila,100);

printf("\nSe ha insertado la pila. Tope Izquierda= %d \t Tope Derecha: %d\n\n", pila.topeizq, pila.topeder);

}

printf("\nPila llena.\n\n");

printf("\nPila quedó con tope izq =%d y tope der =%d ", pila.topeizq, pila.topeder);//pila.A[pila.topeizq]

return 0;

}

============