Universidad Nacional

Autónoma De México

Facultad De Ingeniería

Estructuras De Datos Y Algoritmos I

Actividad Lunes #2:
Algoritmo Usando Pilas

Alumno:

Brandon Hernandez Solis

Fecha:

14/06/2021

Pila.h

```
typedef struct ElementoLista{
char *dato;
struct ElementoLista *siguiente;} Elemento;
typedef struct ListaUbicación{
Elemento *inicio;
int tamaño;} Pila;

/* inicialización */
void inicialización (Pila *tas);

/* Apilar*/
int apilar (Pile *tas, char *dato);

/* Desapilar*/
int desapilar (Pila *tas);

/* Visualización del elemento en la cabeza de la pila */
#define pila_dato(tas)
tas->inicio->dato

/* muestra la pila */
void muestra (Pila *tas);
```

Pila_function.h

```
void inicialización (Pila * tas){
tas->inicio = NULL;
tas->tamaño = 0;}

/* apilar (añadir) un elemento en la pila */
int apilar (Pila * tas, char *dato){
Elemento *nuevo_elemento;
if ((nuevo_elemento = (Elemento *) malloc (sizeof (Elemento))) == NULL)    return
-1;
if ((nuevo_elemento->dato = (char *) malloc (50 * sizeof (char)))      == NULL)
return -1;  strcpy (nuevo_elemento->dato, dato);
nuevo_elemento->siguiente = tas->inicio;
tas->inicio = nuevo_elemento;
tas->tamaño++;
}
```

```
/* desapilar */
int desapilar (Pila * tas){
Elemento *sup_elemento;
if (tas->tamaño == 0)    return -1;
sup_elemento = tas->inicio;
tas->inicio = tas->inicio->siguiente;
free (sup_elemento->dato);
free (sup_elemento);
tas->tamaño--;
return 0;
}

/* visualización de la pila */
void muestra (Pila * tas)
{
Elemento *actual;
int i;
actual = tas->inicio;
for(i=0;i<tas->tamaño;++i)
{
printf("\t\t%s\n", actual->dato);
actual = actual->siguiente;
}
}
```

Pila.c

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include "pila.h"
#include "pila_function.h"
int main () {
Pila *tas;
char *nom;
if ((tas = (Pila *) malloc (sizeof (Pila))) == NULL)    return -1;
if ((nom = (char *) malloc (50 * sizeof (char))) == NULL)    return -1;
inicialización (tas);
printf ("Ingrese una palabra: ");
scanf ("%s", nom);
apilar (tas, nom);
printf ("La pila (%de elementos): \n",tas->tamaño);
printf("\n********** Cabeza de la PILA **********\n");
```

```
muestra(tas);
printf("__________ Bajo de la PILA __________\n\n");
printf ("Ingrese una palabra: ");
scanf ("%s", nom);
apilar (tas, nom);
printf ("La pila (%de elementos): \n",tas->tamaño);
printf("\n********** Cabeza de la PILA **********\n");
muestra(tas);
printf("__________ Bajo de la PILA __________\n\n");
printf ("Ingrese una palabra: ");
scanf ("%s", nom);
apilar (tas, nom);
printf ("La pila (%de elementos): \n",tas->tamaño);
printf("\n********** Cabeza de la PILA **********\n");
muestra(tas);
printf("__________ Bajo de la PILA __________\n\n");
printf ("\nLa ultima entrada (LastInFirstOut) [ %s ] sera eliminada",
pile_dato(tas));  printf ("\nLa ultima entrada es eliminada\n");
desapilar (tas);

/* eliminación del ultimo elemento ingresado */
printf ("La pila (%de elementos): \n",tas->tamaño);
printf("\n********** Cabeza de la PILA **********\n");
muestra(tas);
printf("__________ Bajo de la PILA __________\n\n");
return 0;
}
```