



PROGRAMACION II

MATERIAL para TEORIA PILAS

Código Asignatura 6A4
Año 2023

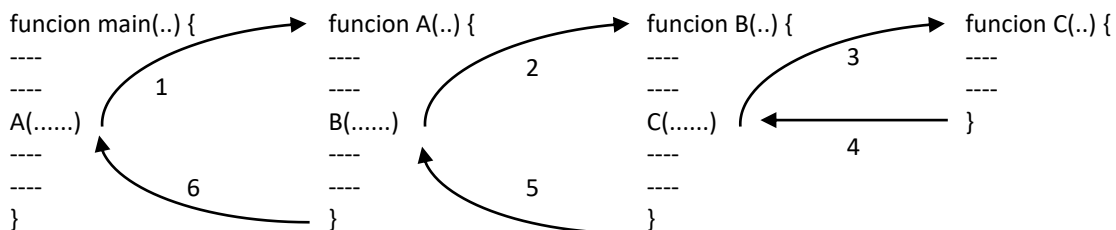
TIPO de DATOS PILA

Una pila o stack es una estructura de datos que consiste en una colección de elementos, todos del mismo tipo, en la cual las operaciones de inserción y eliminación se realizan por un extremo denominado tope (o cima). Sólo el elemento que está en el tope de la pila puede ser consultado o removido y si se agrega un elemento, éste se incorpora sobre el tope.

Un ejemplo típico es “una pila de platos”, en la que ponemos sobre el último apilado y sacamos el que está en el tope (no se incorpora ni se elimina por la base o el medio). También se conoce esta estructura con el nombre de LIFO (Last In First Out), de este modo podemos decir que una pila *invierte* el orden de los elementos que en ella se almacenan. A pesar de su sencillez, es muy adecuada para la información que se comporta de este modo en el almacenamiento y recuperación. Es el caso, por ejemplo de las invocaciones a subprogramas.

Cada subprograma almacena (apila) en el segmento de memoria correspondiente, parámetros, variables locales, temporales y direcciones de retorno.

Por ejemplo, cuando desde un subprograma A se invoca a otro subprograma B, los elementos antes mencionados de B se apilan sobre los de A, quedando en el tope lo concerniente al subprograma activo. Cuando B finaliza, su información se desapila y queda en el tope la información de A, que retoma la ejecución (suspendida por el llamado a B)



TDA PILA

Para definir un Tipo de Dato Abstracto Pila, es necesario definir el tipo y un conjunto de operadores que permitan declarar y utilizar variables del tipo. Para desarrollar algoritmos que almacenen y recuperen elementos de una pila, es necesario conocer las cabeceras de los operadores (interface), pero no los detalles de su implementación (caja negra)

OPERADORES del TDA PILA

Siendo P : pila de objetos de tipo “TElementoP” y x : objeto de tipo “TElementoP”

- ♦ INICIAP(P) → Devuelve en P una pila vacía
- ♦ VACIAP(P) → Devuelve Verdadero si la pila P está vacía, y Falso en caso contrario
- ♦ CONSULTAP(P) → Devuelve el valor del elemento que se encuentra en el tope de la pila P
- ♦ SACAP(P , x) → Devuelve en x el elemento que se encuentra en el tope de la pila removiéndolo de P
- ♦ PONEP(P , x) → Inserta el elemento x en el tope de la pila.

INTERFACE del TDA PILA

```
void IniciaP (TPila * P);
void poneP (TPila * P, TElementoP x);
void sacaP (TPila * P, TElementoP * x);
TElementoP consultaP(TPila P);
int VaciaP (TPila P);
```

UTILIZACION del TDA PILA

Problema: Reescribir un número decimal en su equivalente binario.

```

#include<stdio.h>
#include<pilas.h>

int main()  {
    TPila P;
    TElementoP bit;
    int num, numero;

    IniciaP(&P);
    scanf("%d",&numero);
    num=numero;
    while (numero != 0) {
        poneP(&P, numero % 2);
        numero /= 2;
    }
    printf("Representacion de %d en base 2 \n", num);
    while (!VaciaP(P)) {
        sacaP(&P,&bit);
        printf("%d ",bit);
    }
    return 0;
}

```

IMPLEMENTACION del TDA PILA**⇒ IMPLEMENTACION ESTATICA**

```

#define MAX 50
typedef int TElementoP;
typedef struct {
    TElementoP datos[MAX];
    int tope; } TPila;

void poneP(TPila *P, TElementoP x) {
    if ( ((*P).tope) != MAX-1)
        (*P).datos[++((*P).tope)] = x;
}

void sacaP(TPila *P, TElementoP* x)  {
    if ((*P).tope) != -1)
        *x = (*P).datos[((*P).tope)--];
}

TElementoP consultaP(TPila P)  {
    if ((P.tope) != -1)
        return P.datos[P.tope];
}

int VaciaP(TPila P)  {
    return (P.tope == -1);
}

void IniciaP (TPila *P) {
    (*P).tope=-1;
}

```

⇒ IMPLEMENTACION DINAMICA

```
typedef int TElementoP;
typedef struct nodop {
    TElementoP dato;
    struct nodop *sig; } nodop;
typedef nodop *TPila;

void poneP(TPila *P, TElementoP x) {
    TPila N;
    N = (TPila)malloc(sizeof(nodop));
    N->dato = x;
    N->sig = *P;
    *P=N;
}

void sacaP(TPila *P, TElementoP * x) {
    TPila N;
    if (*P) {          // if (*P != NULL)
        N = *P;
        *x = (N)->dato;
        *P = (N)->sig;
        free(N);
    }
}

TElementoP consultaP(TPila P) {
    if (P)              // if (P != NULL)
        return P->dato;
}

int VaciaP(TPila P) {
    return (P == NULL);
}

void IniciaP(TPila *P) {
    *P =NULL;
}
```