UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Estructura de Datos y Algoritmos I

Actividad Asíncrona #02 Lunes

Algoritmo Operación Push y Pop

Alumno:

Flores Ramírez Eduardo Amilcar

14 de junio del 202

Algoritmo para la operación Push y Pop.

Antes de comenzar a trabajar en el algoritmo para la creación de las operaciones Push y Pop, primero debemos de conocer el algoritmo de implementación de una pila ya que estas operaciones normalmente se trabajan sobre este tipo de almacenamiento.

Paso 1: Implementación de las Pilas.

Una pila está conformada por dos elementos:

Primero debemos conocer si en nuestra computadora cuanta con el espacio suficiente para almacenar los elementos insertados en la pila.

Siguiendo de esto, es importante definir un nuevo tipo estructura llamado "stack" con item: un arreglo de 0 a “máximos” elementos enteros top : un numero de -1 a (máximos – 1).

Es posible escribir un código en C/C++ que represente lo anteriormente propuesto:

```
// En la parte de definiciones
#define maxElem 100

// En la parte de tipos
struct stack {
  int item[maxElem];
  int top;
};

// En la parte de variables
struct stack A;
```

Una pila es una colección ordenada de objetos.

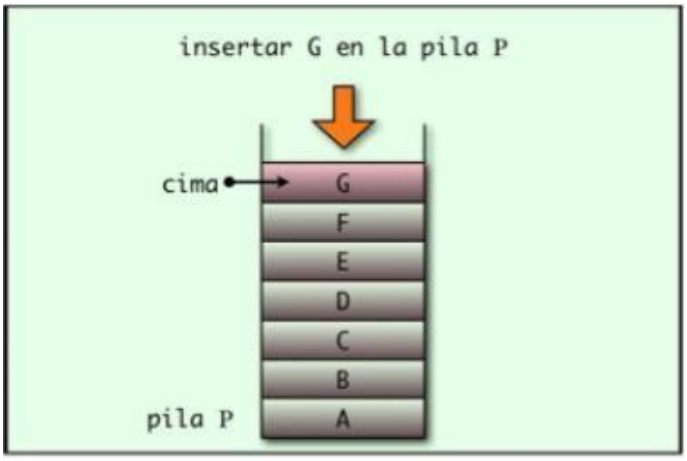
• En C, los arreglos permiten almacenar colecciones ordenadas.

• La desventaja de implementar una pila mediante un arreglo es que esta última es de tamaño fijo, mientras que usando punteros la pila puede ser de tamaño dinámico.

Una vez creada la pila, podemos rellenarla con datos de nuestro interés, con la ayuda de la operación push.

Paso 2: Algoritmo Push.

Existe solamente un lugar en donde cualquier elemento puede ser agregado a la pila. Después de haber insertado el nuevo elemento, G ahora es el elemento en la cima.



La Operación Push

Ahora bien, esta operación sirve para insertar un elemento e en la pila S, y se va a representar de la siguiente manera, se va a escribir: push (S,e).

Una vez realizada esta operación sucede lo siguiente, el elemento en la cima de la pila S ahora es e, de tal modo que la consola lo interpreta de la siguiente manera:

Es posible escribir un código en C/C++ que represente lo anteriormente propuesto:

Primero la operación push recibe : la dirección de una estructura pila y un elemento entero:

```
void push(struct stack *S,int e){
```

Después se debe de incrementa el tope (cima) de la pila para agregar el elemento en una posición libre de la pila:

```
S->top++;
```

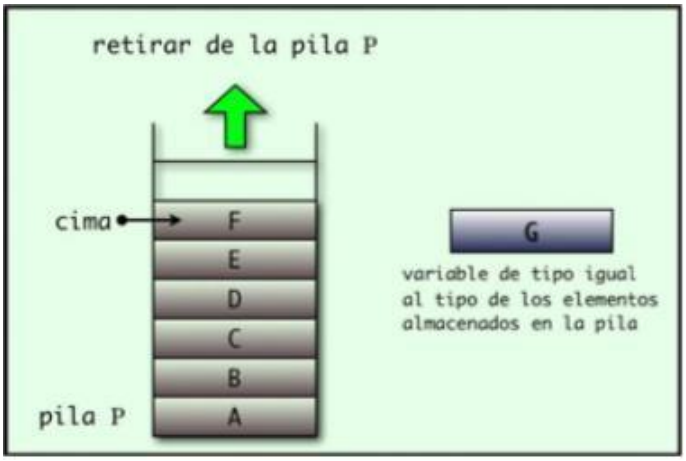
Y finalmente se asigna el valor e, en la casilla S->top:

```
S->item[S->top]=e;
```

De este modo se logra almacenar un nuevo dato en la cima de nuestra pila.

Paso 3: Algoritmo Pop.

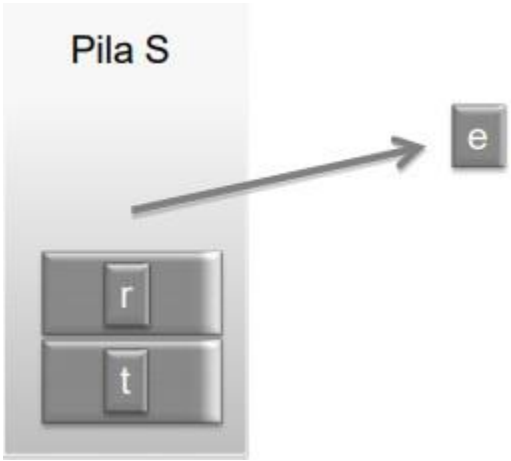
Basta indicar que sea retirado un elemento. No podemos decir “retirar C” , porque C no está en la cima de la pila.



La operación pop:

Finalmente, una vez creada la pila y declarados los datos, podremos retirar los elementos que se encuentran en la cima de la pila S, y se va a representar de la siguiente manera: pop (S,e).

Al hacer esto la función devuelve un tipo entero al recibir la dirección de una variable de tipo estructura pila (struct stack *). Para finalmente disminuir el tope de la pila.



Es posible escribir un código en C/C++ que represente lo anteriormente propuesto:

La función devuelve un tipo entero al recibir la dirección de una variable de tipo estructura pila (struct stack *):

```
int pop(struct stack *S){
```

Esto generara que se elimine el valor al ser devuelto y se decrementa el tope de la pila:

```
valReturn=S->item[S->top];
S->top--;
```

De este modo se logra borrar un dato almacenado en la cima de la pila.

Referencias:

Cumplido, R., & Rodríguez Flores, L. (2015). Pilas y Colas. Recuperado 14 de junio de 2021, de https://ccc.inaoep.mx/ingreso/programacion/corto2015/Curso-PROPE-PyED-5-Pilas-Colas.pdf