

Listas, Pilas y Colas

Bernal Chauayo, Luis Antonio
Lacuaña Apaza, Margarita
Mendoza Villarroel, Alexis
Villena Zevallos, Ademir

Ciencia de la Computación
Universidad Nacional de San Agustín

21 de Octubre del 2016

Índice

1 Lista Simple

- Definición
- Objetos
- Insertar
- Eliminar

2 Lista Circular

- Definición

3 Pilas

- Definición

- Push

- Pop

- Push - Lista

- Pop - Lista

4 Colas

- Introducción

- Encolar-Push

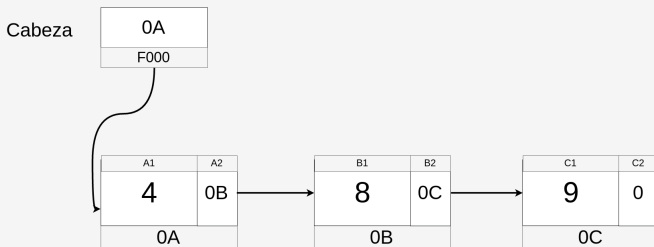
- Desencolar-Pop

- Frente-Front

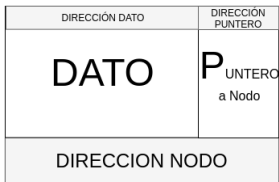
5 Bibliografía

Lista simple

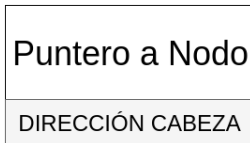
- Una lista enlazada simple es una estructura de datos formada por nodos, en la que cada nodo apunta al siguiente. De este modo guardando una referencia a la cabeza de la lista podemos acceder a todos los elementos de la misma.



- Nodo



- Lista

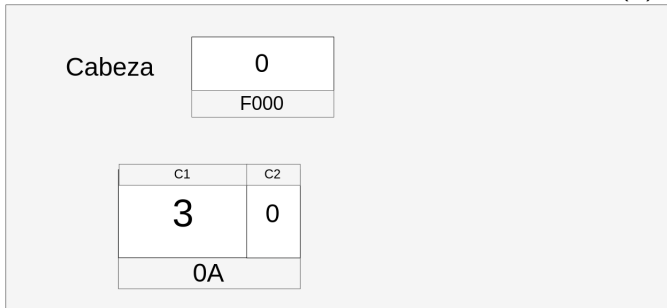


Cabeza

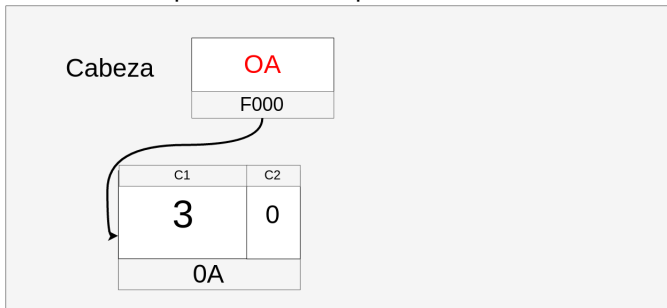
0

F000

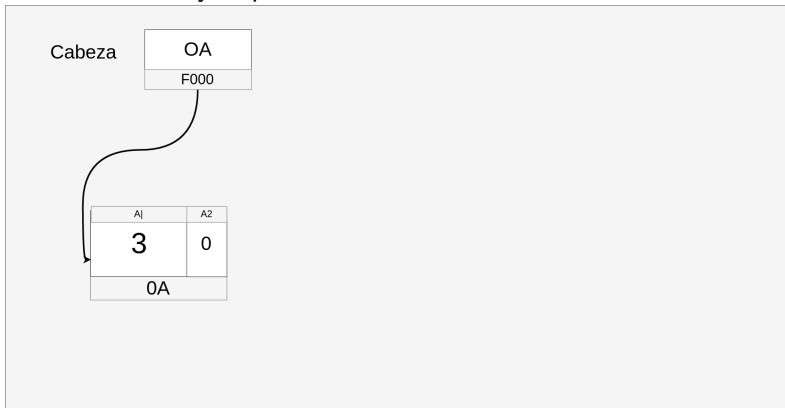
1.- Creamos un nuevo nodo con el dato a insertar (3).



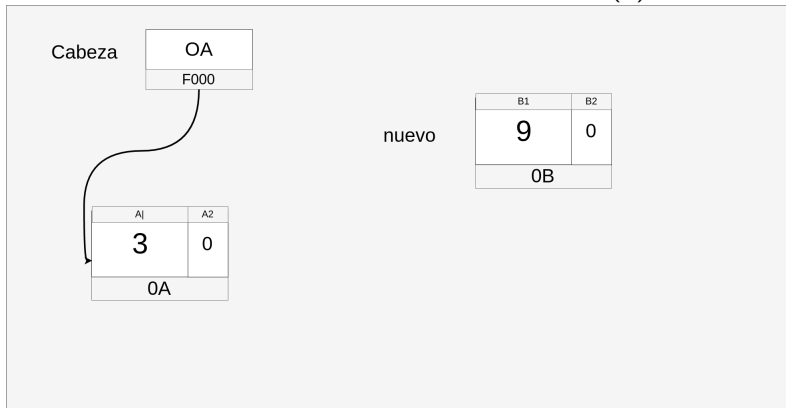
2.- Hacemos que la cabeza apunte al nuevo nodo.



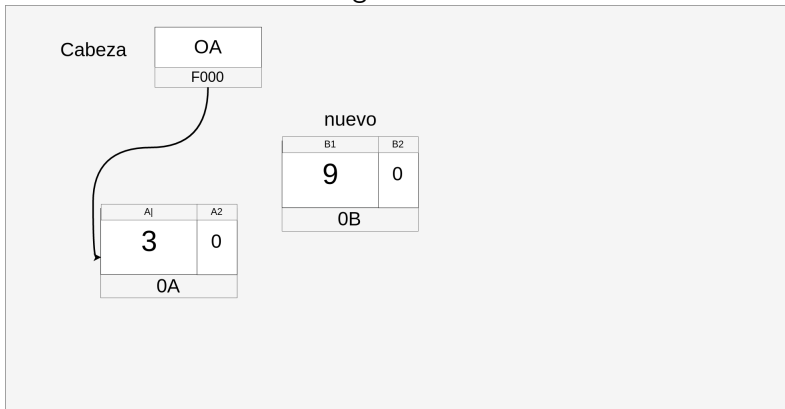
Inserción cuando el puntero a la cabeza es diferente de 0 y el dato a insertar es mayor que el dato de la cabeza



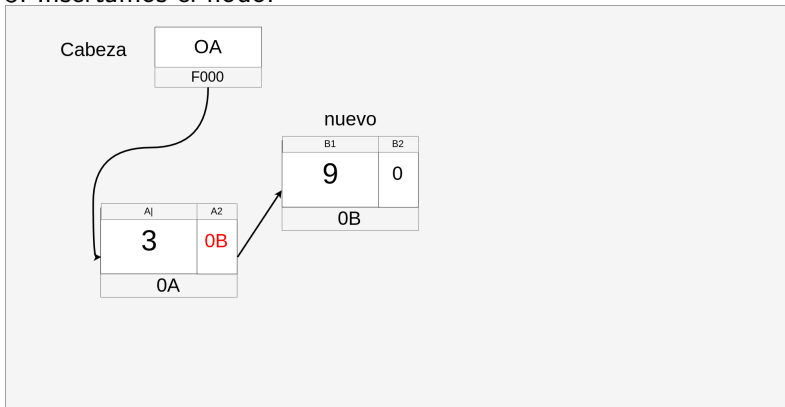
1.-Creamos un nuevo nodo con el dato a insertar (9).



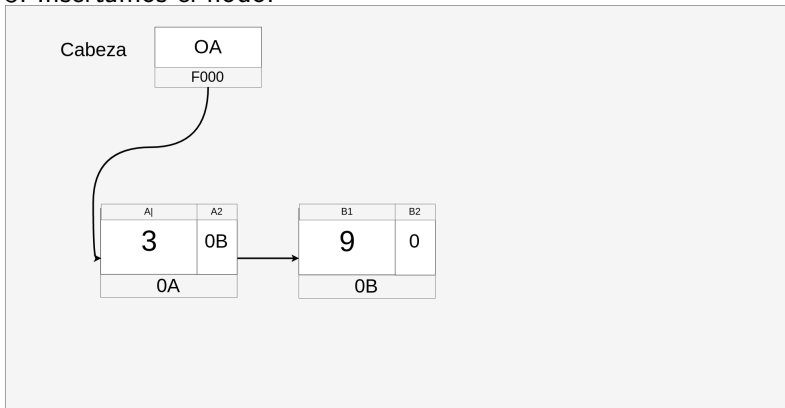
2.-Ubicamos el nodo en el lugar donde se insertara.



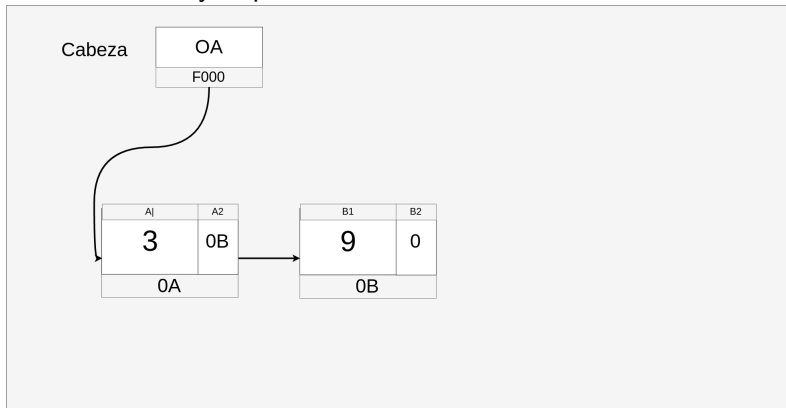
3.-Insertamos el nodo.



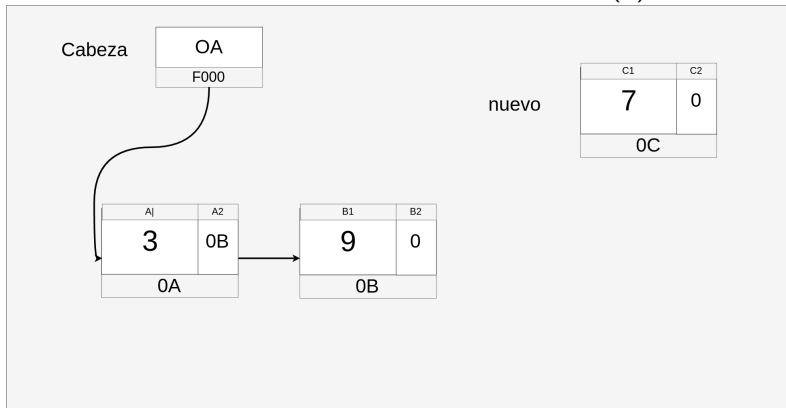
3.-Insertamos el nodo.



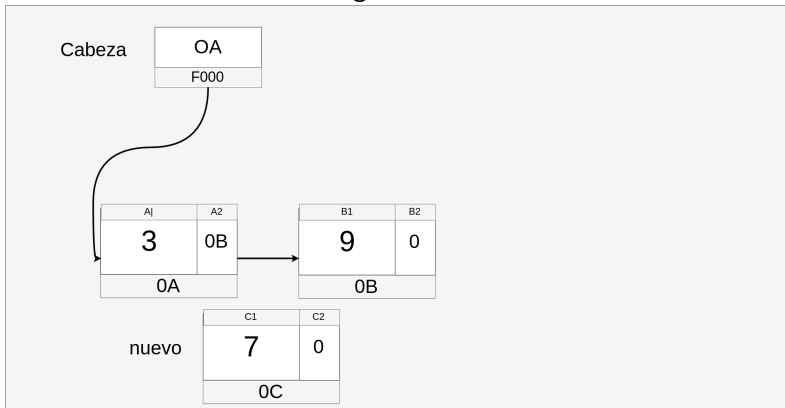
Inserción cuando el puntero a la cabeza es diferente de 0 y el dato a insertar es mayor que el dato de la cabeza



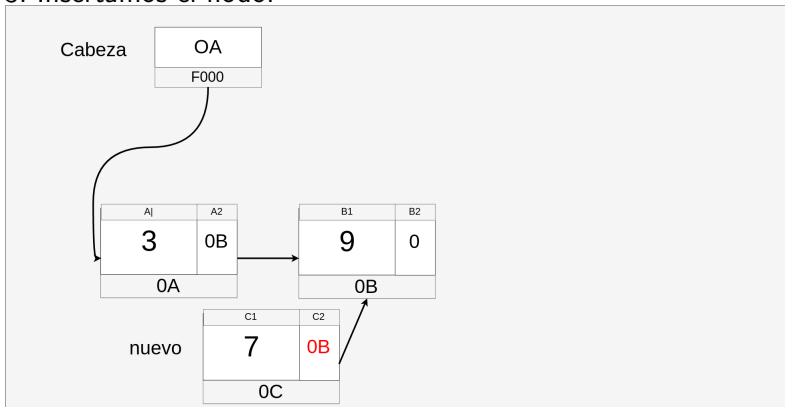
1.-Creamos un nuevo nodo con el dato a insertar (7).



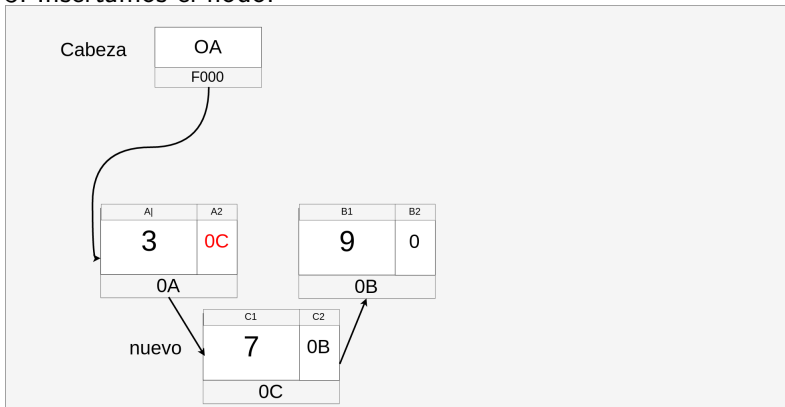
2.-Ubicamos el nodo en el lugar donde se insertara.



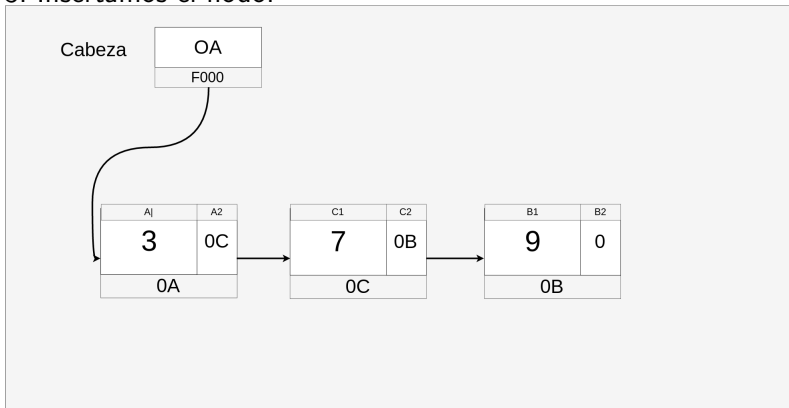
3.-Insertamos el nodo.



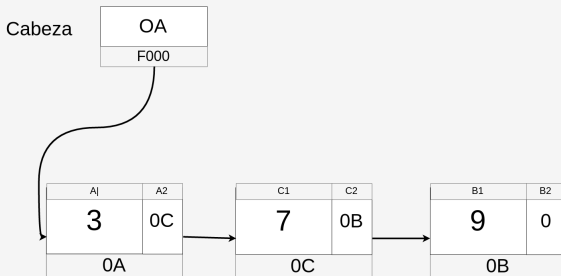
3.-Insertamos el nodo.



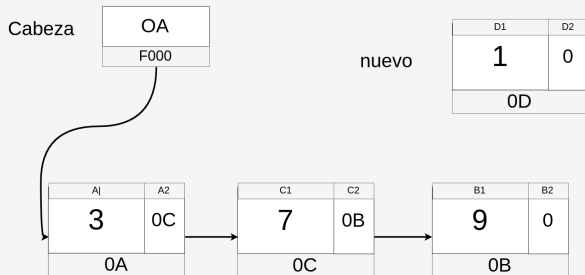
3.-Insertamos el nodo.



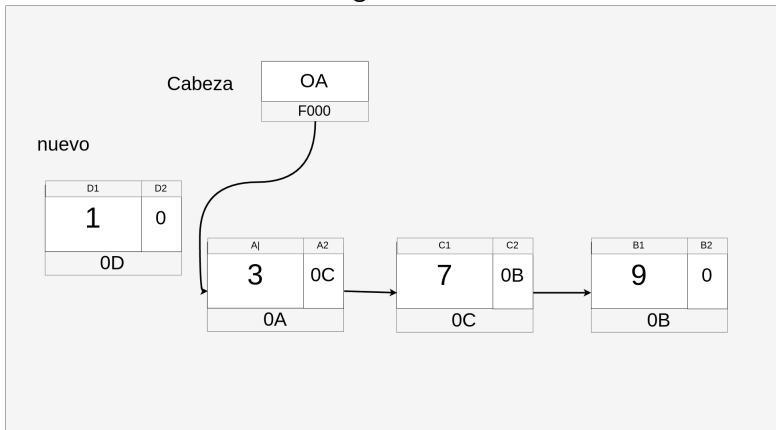
Insertión cuando el puntero a la cabeza es diferente de 0 y el dato a insertar es menor que el dato de la cabeza



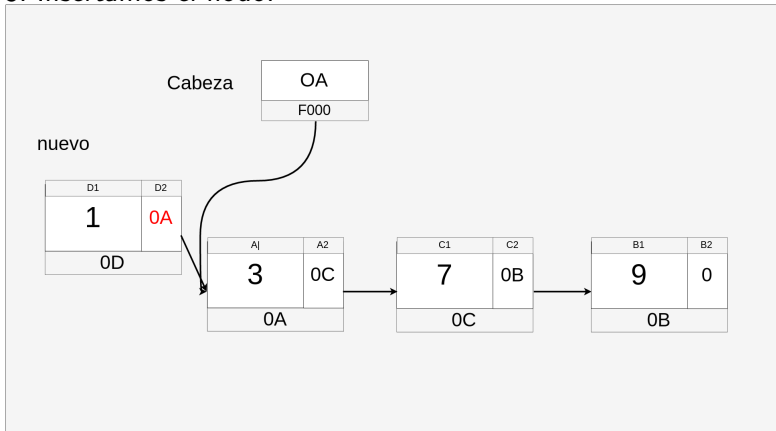
1.-Creamos un nuevo nodo con el dato a insertar (1).



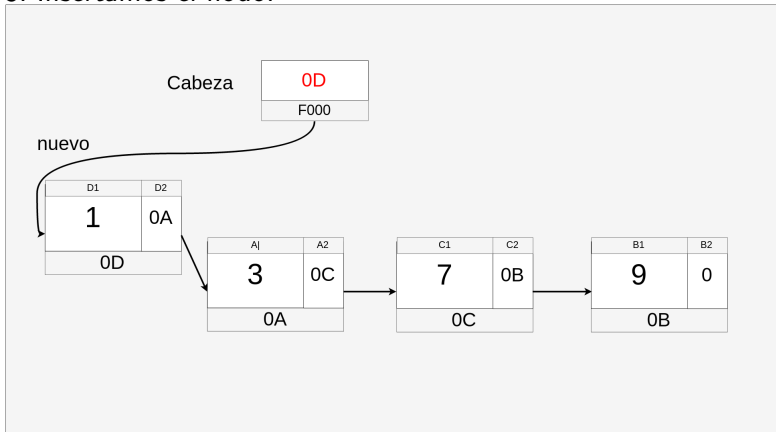
2.-Ubicamos el nodo en el lugar donde se insertara.



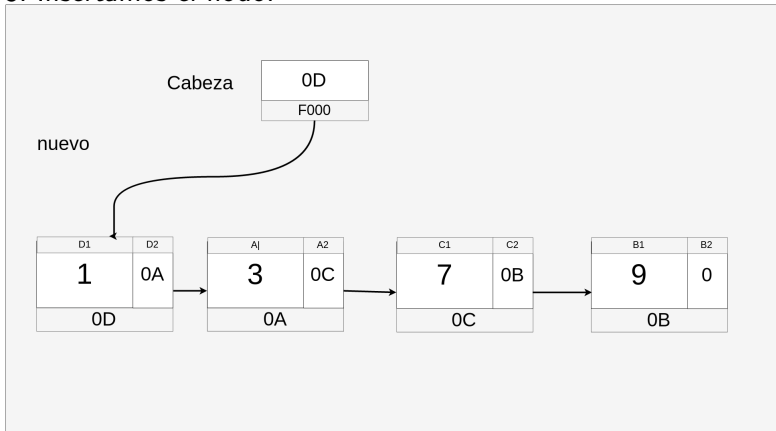
3.-Insertamos el nodo.



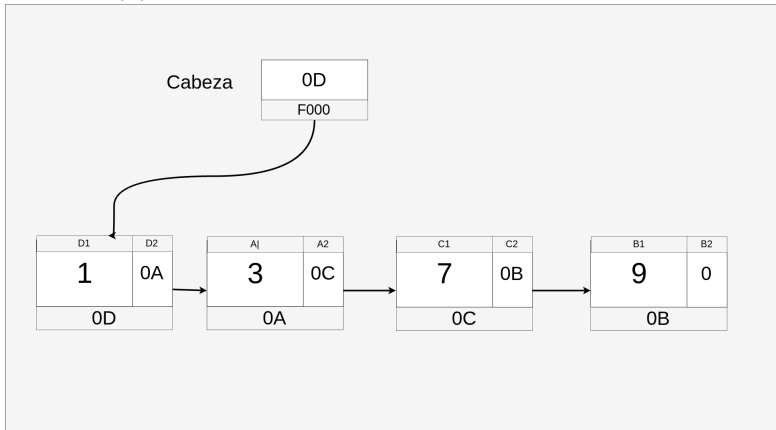
3.-Insertamos el nodo.



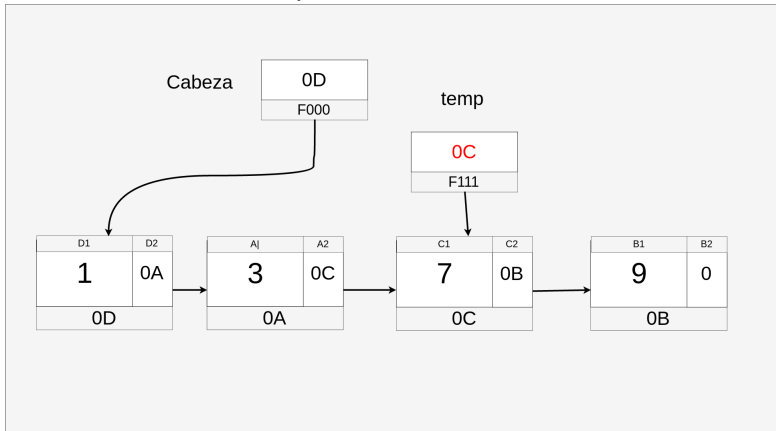
3.-Insertamos el nodo.



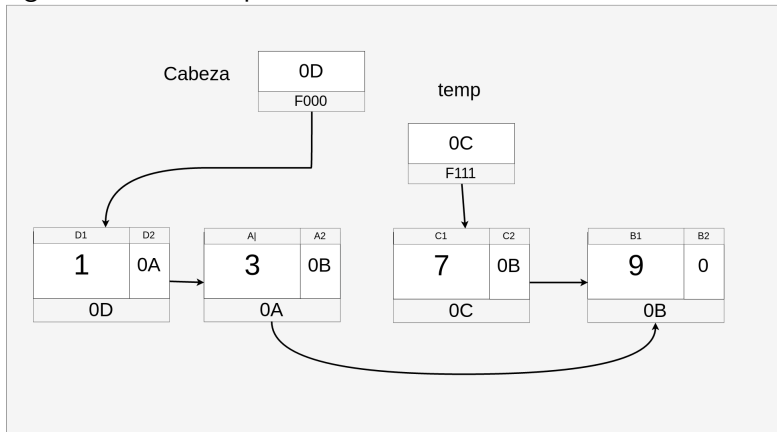
Eliminar (7).



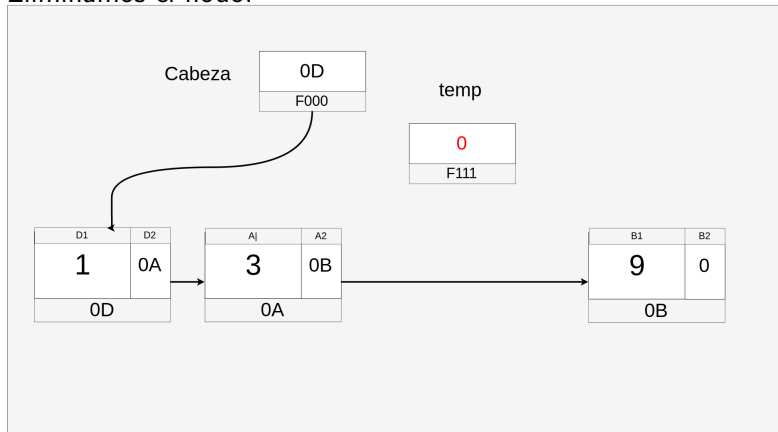
Seleccionamos el nodo que vamos a eliminar.



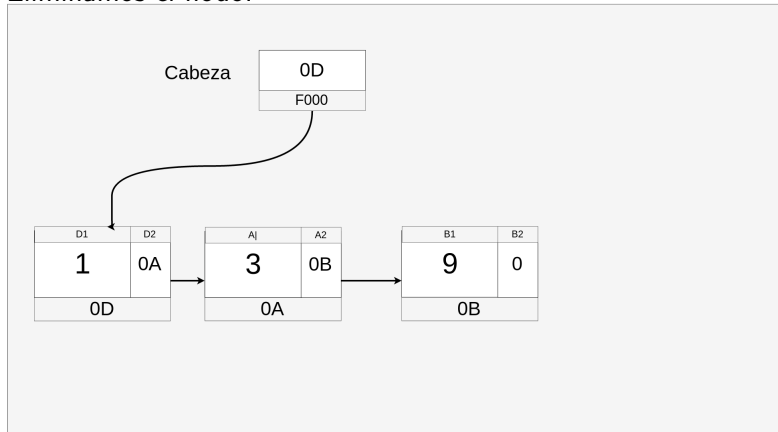
El nodo que apuntaba al nodo que eliminaremos ahora apuntara al siguiente del nodo que eliminaremos.



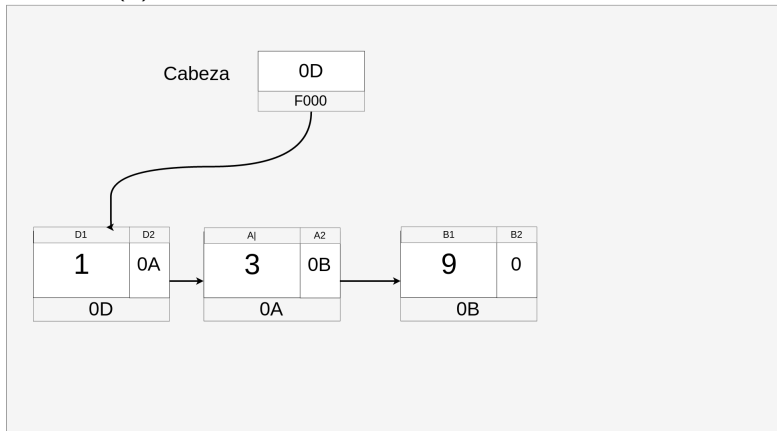
Eliminamos el nodo.



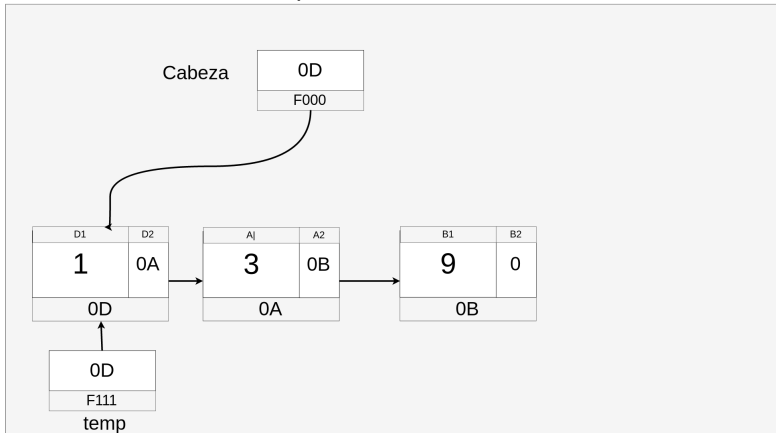
Eliminamos el nodo.



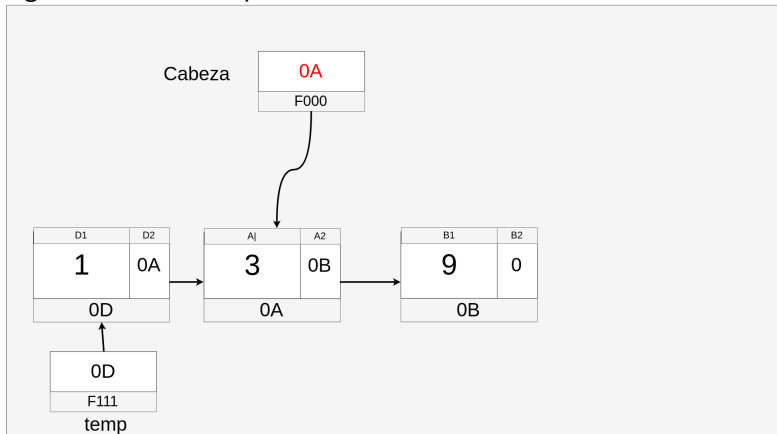
Eliminar (1).



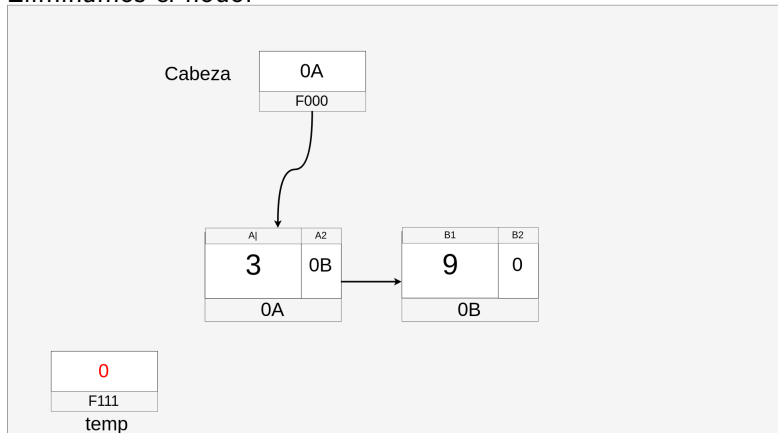
Seleccionamos el nodo que vamos a eliminar.

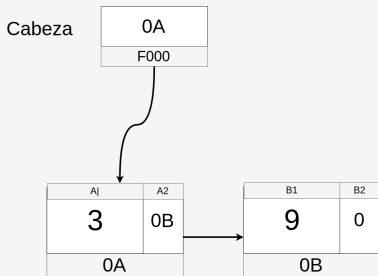


El nodo que apuntaba al nodo que eliminaremos ahora apuntara al siguiente del nodo que eliminaremos.



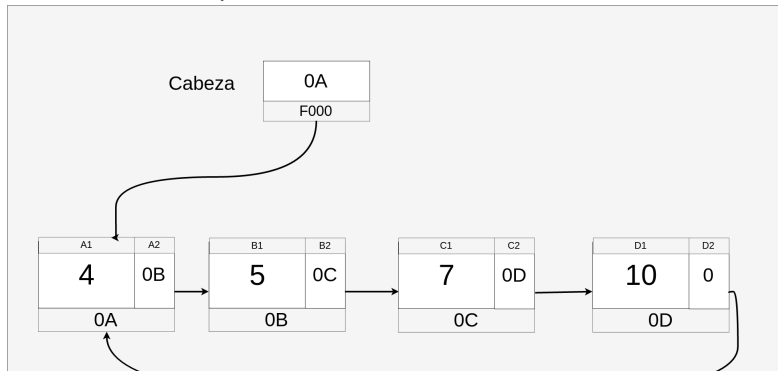
Eliminamos el nodo.





Lista Circular

- Una Lista Circular es una lista simple en la que el ultimo nodo apunta al primer nodo.
- Las operaciones de insertar y eliminar son similares a las de una lista simple.



Push

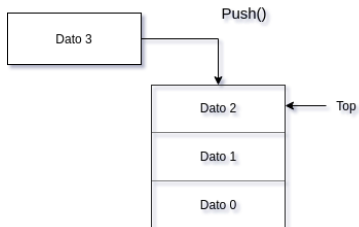


Figure: Inserción de un elemento

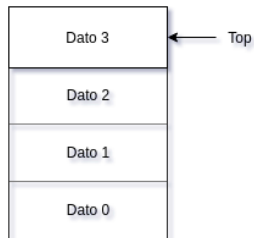


Figure: Pila despues de la inserción

Pop

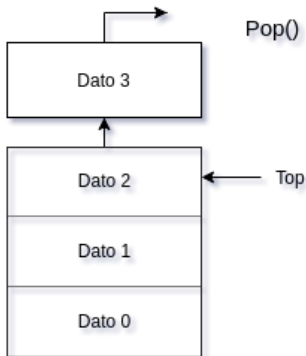


Figure: Eliminación de un elemento

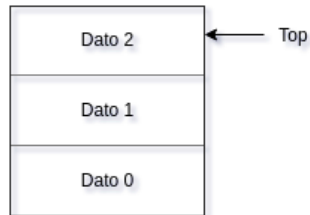


Figure: Pila despues de eliminar un elemento

Implementacion Clase Nodo

```
1  template <class T>
2
3  class Nodo
4  {
5
6  public:
7
8      T          m_dato;
9      Nodo<T> *  m_psiguiente;
10
11 public:
12
13     Nodo(T d){
14         m_psiguiente = 0;
```

Implementacion Clase Pila

```
1  #include "Nodo.h"
2  #include <ostream>
3
4  using namespace std;
5
6  template <class T>
7  class Pila
8  {
9  public:
10         Nodo<T> *    m_phead;
11
12  public:
13         Pila(){
14                 m_phead = 0;
```


Push - Lista

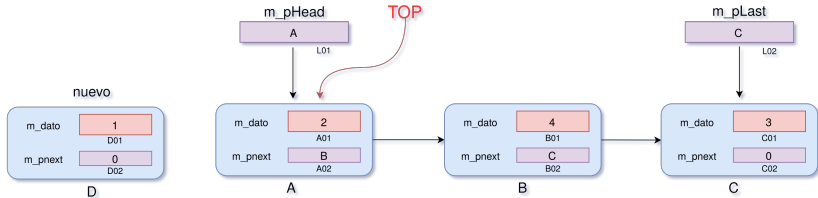


Figure: Crear nuevo Nodo

Push - Lista

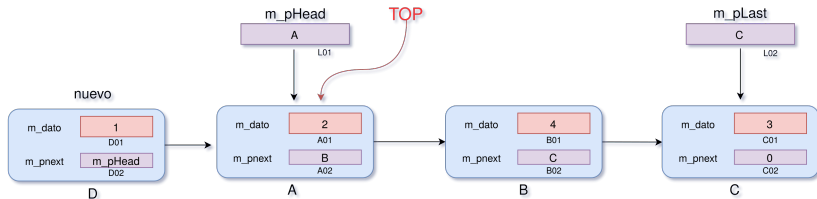


Figure: Apuntar el Siguiete del Nuevo a la Cabeza de la Lista

Push - Lista

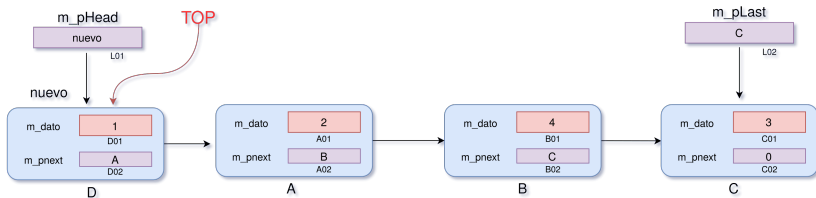


Figure: Apuntar la Cabeza al Nuevo

Push - Lista

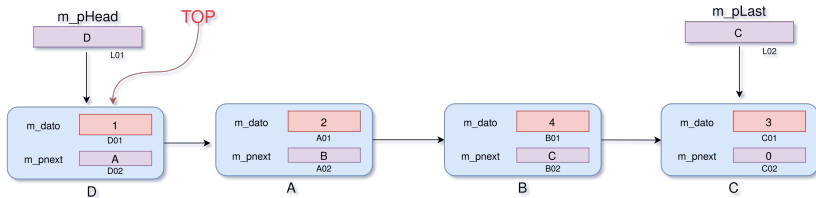


Figure: Resultado

Push

```
1  template <class T>
2  void Pila<T>::push(T d){
3      Nodo<T> * nuevo = new Nodo<T>(d);
4      if(!m_phead)
5          m_phead = nuevo;
6      else{
7          nuevo->m_psiguiente = m_phead;
8          m_phead = nuevo;
9      }
10 }
```

Pop - Lista

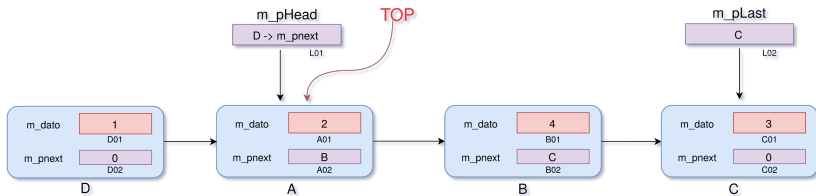


Figure: Apuntar la cabeza de la lista al siguiente de la cabeza

Pop - Lista

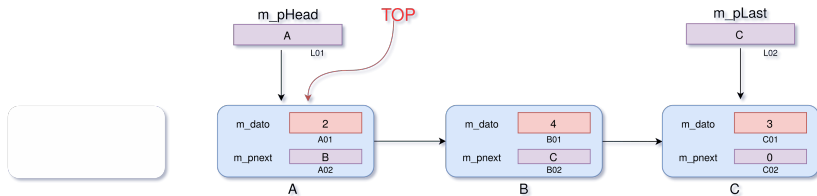


Figure: Borrar el nodo y retornar el dato

Pop

```
1  template <class T>
2  T Pila<T>::pop(){
3      Nodo<T> * temp = m_phead;
4      m_phead = m_phead->m_psiguiente;
5      T dato = temp->m_dato;
6      delete(temp);
7      return dato;
8  }
```


Print

```
1  template <class T>
2  void Pila<T>::print(ostream & os){
3      Nodo<T> * temp = m_phead;
4      while(temp){
5          os<<temp->m_dato<<"->";
6          temp = temp->m_psiguiente;
7      }
8  }
```


Encolar-Push

Se añade un elemento al final de esta. Si la cola esta vacía, el elemento es tanto la cabeza como la cola.

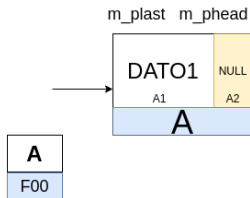


Figure: Insertar caso 1

Cuando insertamos un elemento y la cola no esta vacía, este se añade al final de la cola.
Por último actualizamos el último elemento.

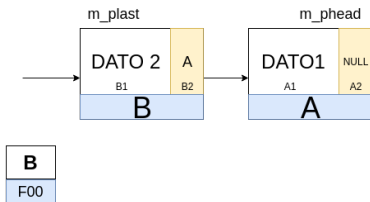


Figure: Insertar caso 2

Desencolar-Pop

Se elimina el elemento frontal de la cola, el elemento que ingreso primero. .

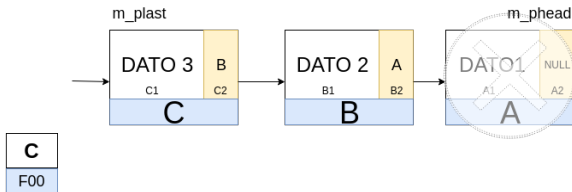


Figure: Eliminar

Desencolar-Pop

Y actualizamos el primer elemento. .

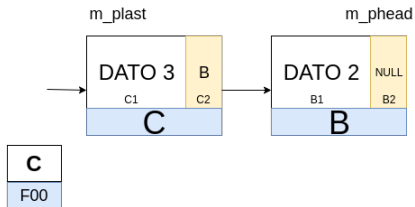


Figure: Actualizar al eliminar

Frente-Front

Se devuelve el elemento frontal de la cola, es decir, el primer elemento que entró.

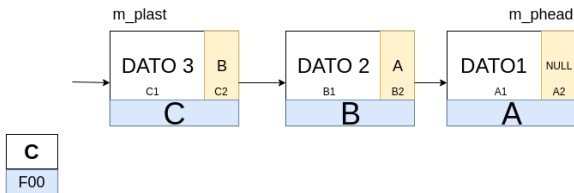


Figure: Devolver elemento frontal

Frente-Front

. .

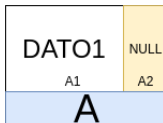


Figure: Resultado

Bibliografía



Lopez C.[2016].*En Algoritmos y Estructuras de Datos*.
Universidad Nacional de San Agustín.