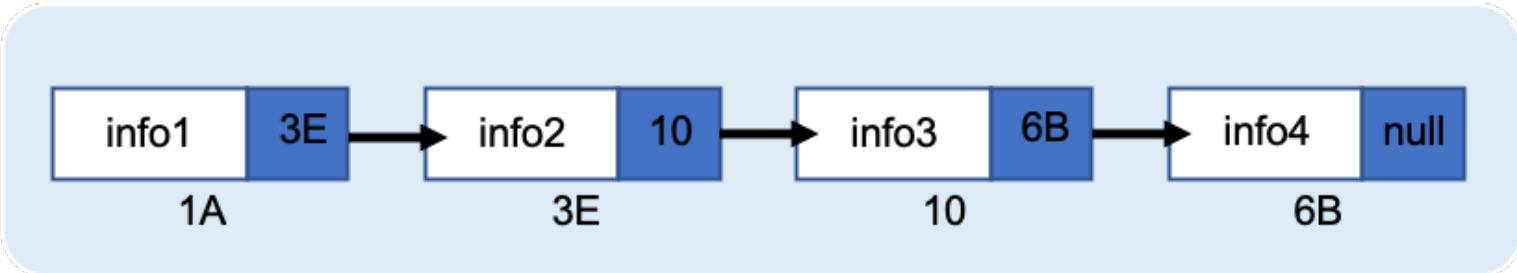
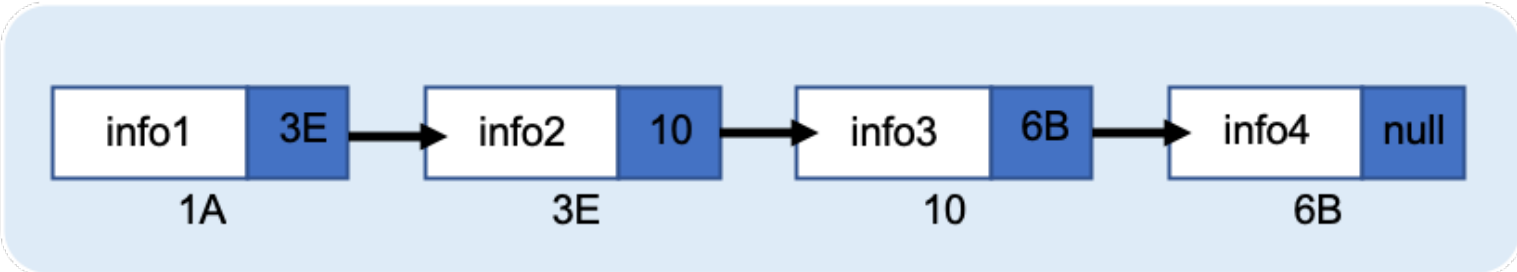
Lista enlazada

- Conjunto de nodos conectados entre sí, situados en direcciones de memoria no consecutivas.
- Cada nodo se compone de una sección de datos y una referencia al siguiente nodo de la lista.



Lista enlazada



- Para insertar un elemento cualquiera debemos ir recorriendo la lista.
- Ventajas con respecto a vectores: una inserción en medio de la lista no requiere mover todos los elementos que se encuentran después del punto de inserción, mientras que en un vector es necesario recorrer todos los elementos para abrir espacio al nuevo elemento.
- Si se permite el acceso a la lista sólo por el primer elemento, entonces se comporta como una **pila**.
- Si las inserciones se realizan por el último elemento, y los accesos sólo por el inicio, entonces se comporta como una **cola**.

Lista enlazada: operaciones básicas

- Insertar elementos
- Mostrar elementos
- Eliminar elementos

Lista enlazada: implementación para pilas (1/5)

```c
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

struct Nodo
{
    int dato;
    struct Nodo *ant;
};

int vacia(struct Nodo *tope)
{
    return (!tope)? 1:0;
}

struct Nodo* crearNodo(int n)
{
    //PASO 1: Creamos el nuevo nodo
    struct Nodo *nuevo_nodo;
    nuevo_nodo = (struct Nodo*)malloc(sizeof(struct Nodo));
    //PASO 2: Asignar el dato al nuevo nodo
    nuevo_nodo->dato = n;
    nuevo_nodo->ant = NULL;
    return(nuevo_nodo);
}
```

Lista enlazada: implementación para pilas (2/5)

```
struct Nodo* push(struct Nodo *tope, int n)
{
    struct Nodo *aux;
    aux = crearNodo(n);
    aux->ant = tope;
    tope = aux;
    return(tope);
}

void mostrar(struct Nodo *tope)
{
    struct Nodo *aux;
    if(!tope)
        printf("La pila esta vacia");
    else
    {
        aux = tope;
        do{
            printf("%d ", aux->dato);
            aux = aux->ant;
        }while(aux != NULL);
    }
    getch();
}
```

Lista enlazada: implementación para pilas (3/5)

```
struct Nodo* pop(struct Nodo *tope)
{
    struct Nodo *aux;
    int dato;
    aux = tope;
    tope = tope->ant;
    free(aux);
    return(tope);
}
```

Lista enlazada: implementación para pilas (4/5)

```
void main()
{
    struct Nodo *tope = NULL;
    int dato, op;
    do{
        printf("PROGRMA QUE IMPLEMENTA PILAS CON LISTAS ENLAZADAS\n\n");
        printf("1. Push\n");
        printf("2. Pop\n");
        printf("3. Mostrar\n");
        printf("4. Salir\n");
        printf("Opcion: ");
        scanf("%d", &op);
        switch(op)
        {
            case 1:   printf("Introduce un numero: ");
                      scanf("%d", &dato);
                      if(tope == NULL)
                          tope = crearNodo(dato);
                      else
                          tope = push(tope,dato);
                      break;
```

Lista enlazada: implementación para pilas (5/5)

```
            case 2:   if(!vacia(tope))
                        {
                            printf("El dato eliminado es: %d\n", tope->dato);
                            tope = pop(tope);
                            printf("El nuevo dato en la cima es: %d\n", tope->dato);
                        }
                        else
                            printf("Pila vacia\n");
                        getch();
                        break;
            case 3:   if(!vacia(tope))
                            mostrar(tope);
                        else printf("Pila vacia\n");
                        break;
            case 4:   break;
        }

    }while(op != 4);
}
```