Estructuras dinámicas de datos

Estructuras Estáticas

- **La memoria es reservada, o asignada, en el momento de la compilación del programa.**

- **Tamaño delimitado: ocupan un número fijo de posiciones de memoria.**

Estructuras Dinámicas de datos

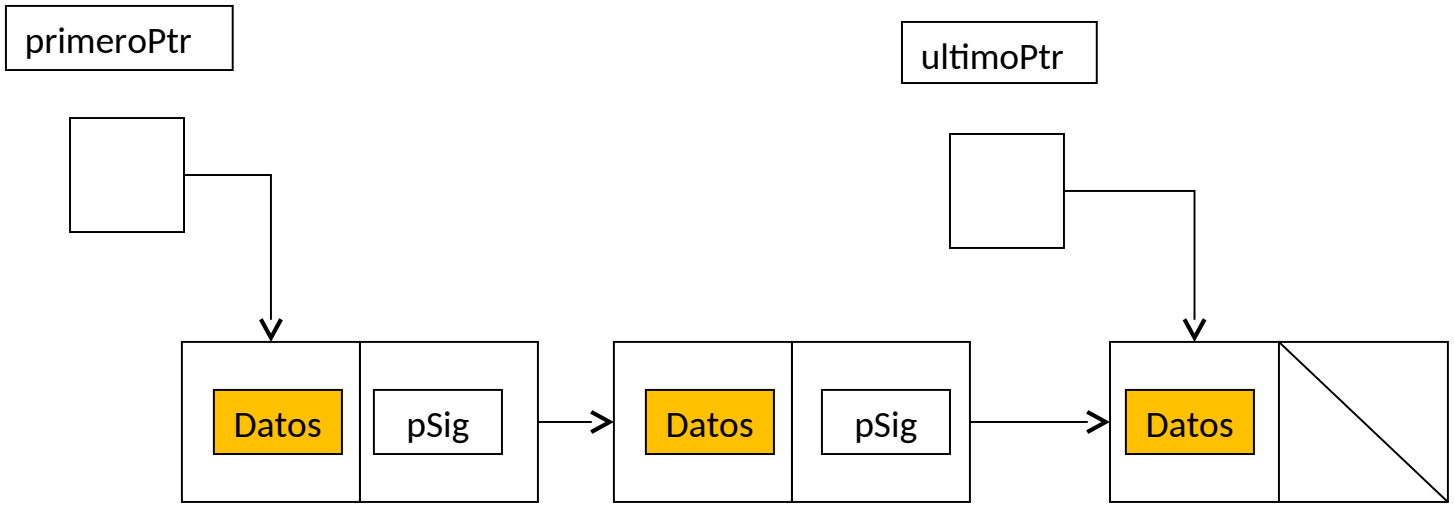
- **Se les asigna memoria en tiempo de ejecución.**
- **Cambian su tamaño, al poder expandirse y contraerse.**
- **Se crean en tiempo de ejecución del programa invocando a la función "malloc".**
- **Con la función “free” se libera el espacio de los miembros generados con “malloc”.**

Tipos: Estructuras Dinámicas

- **Listas Enlazadas.**
- **Pilas.**
- **Colas.**

Listas Enlazadas

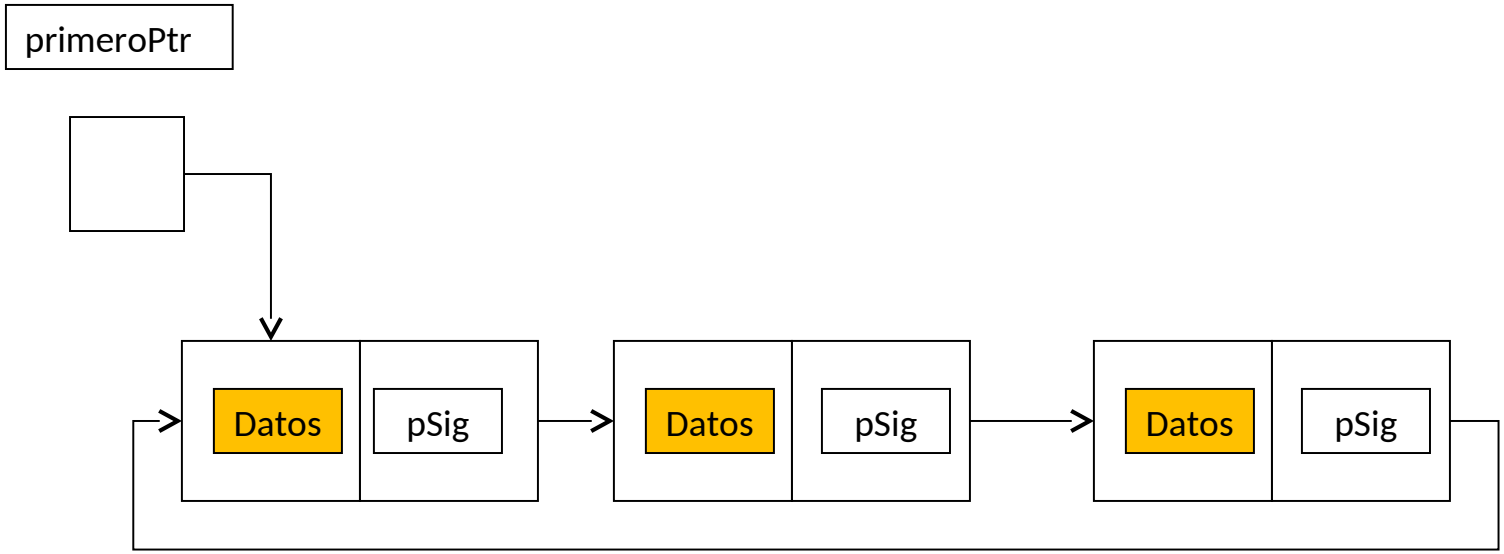
Representación gráfica de una lista simple



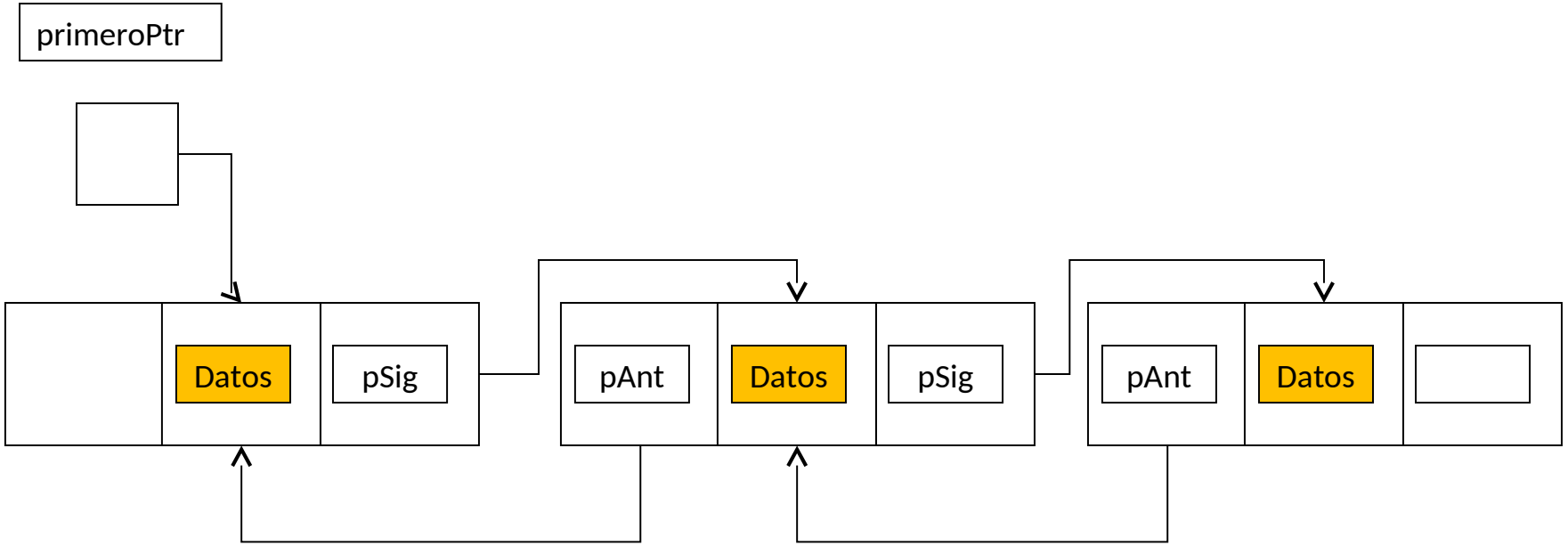
Datos: Son los datos que contiene cada nodo.

pSig: es un puntero al siguiente nodo

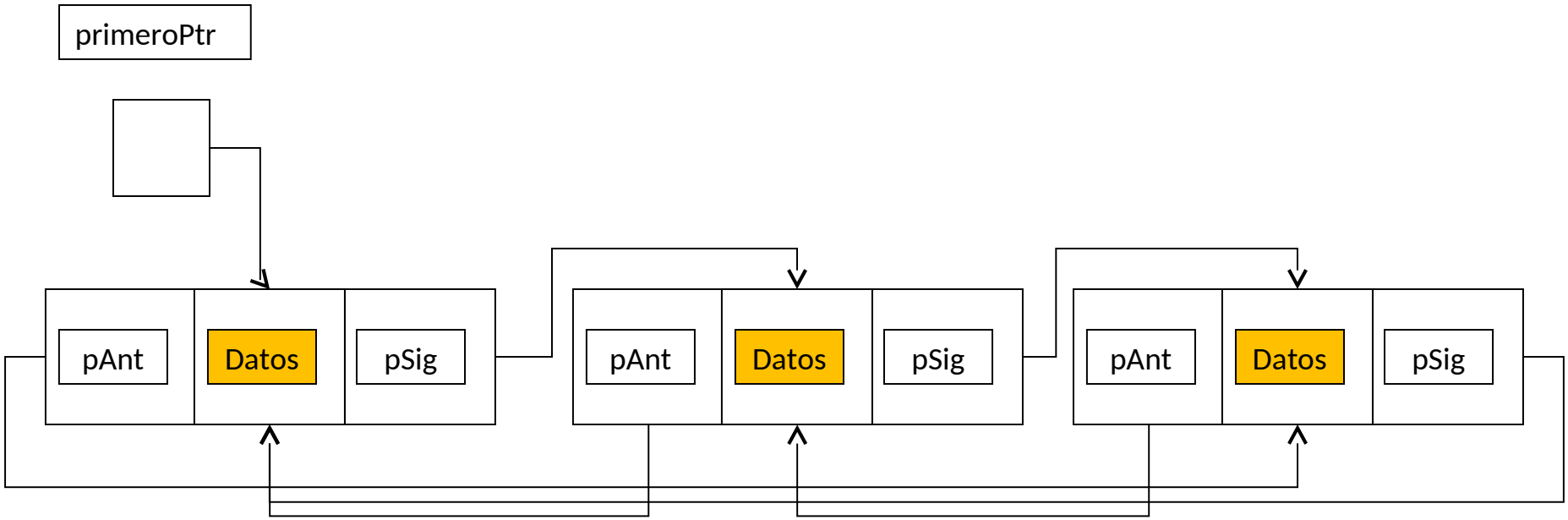
Listas circulares con enlace simple



Listas enlazadas con enlace doble

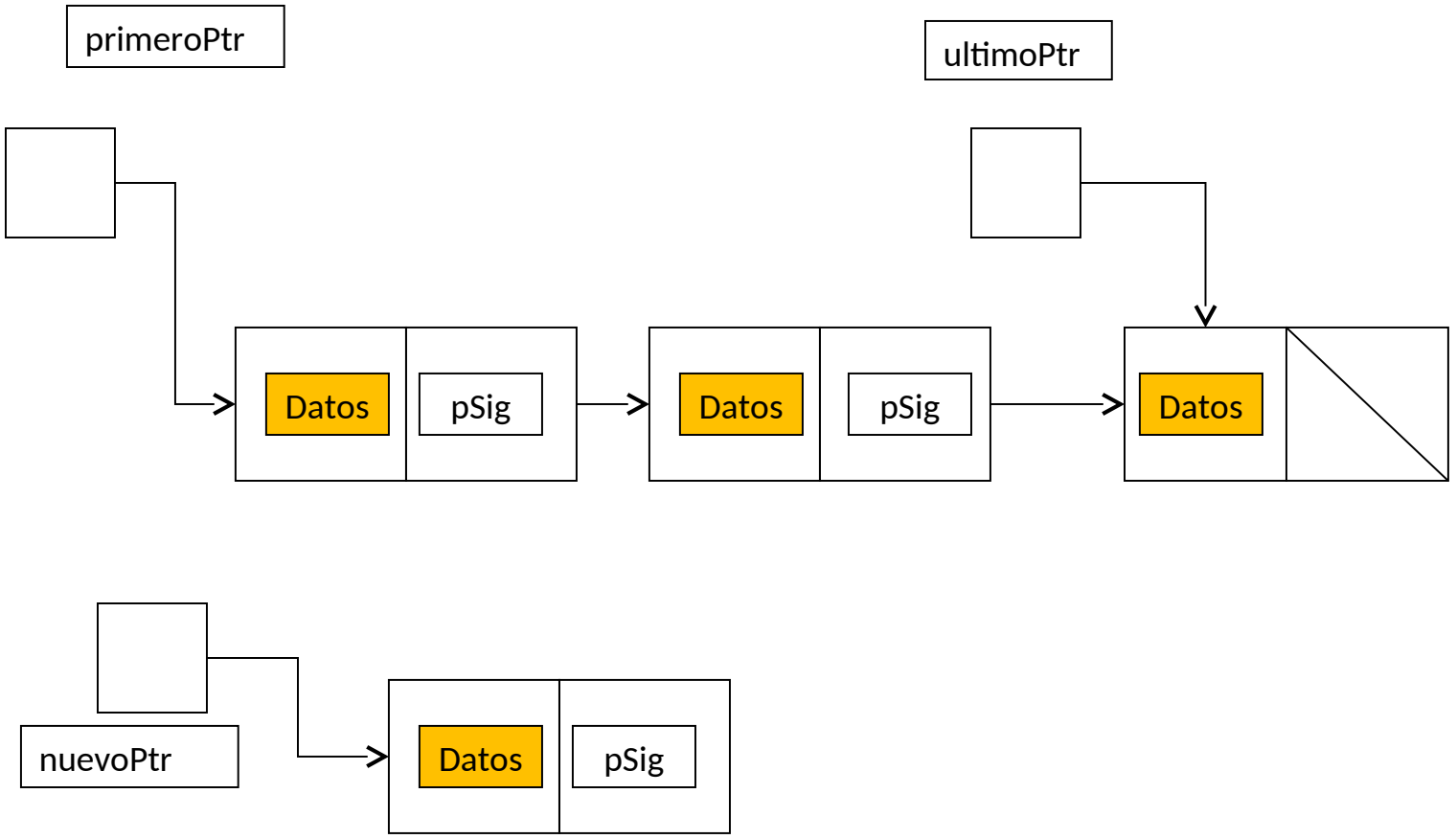


Listas circulares con enlace doble



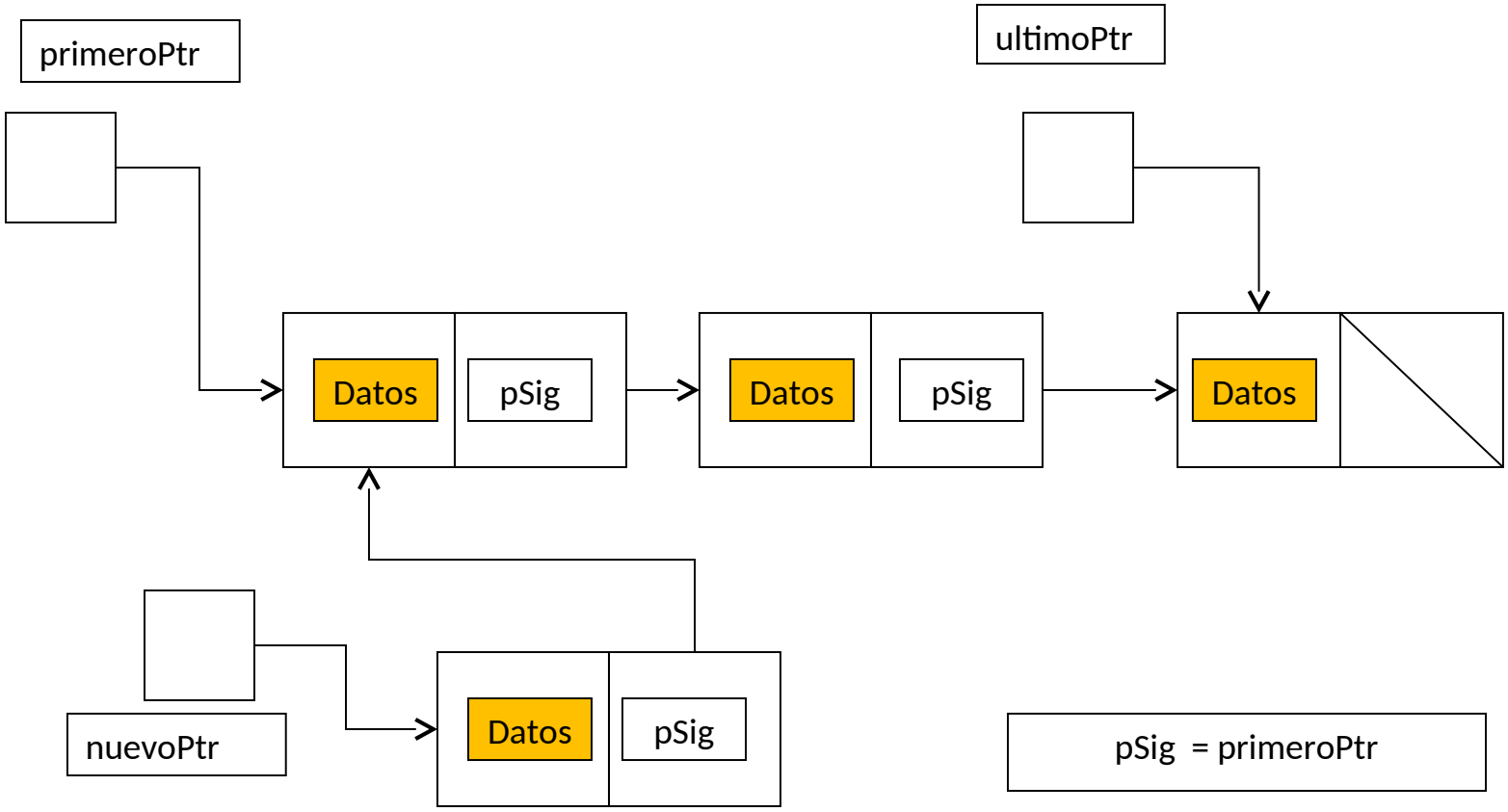
Lista simple operación: Insertar al frente

Paso 1: Crear nodo



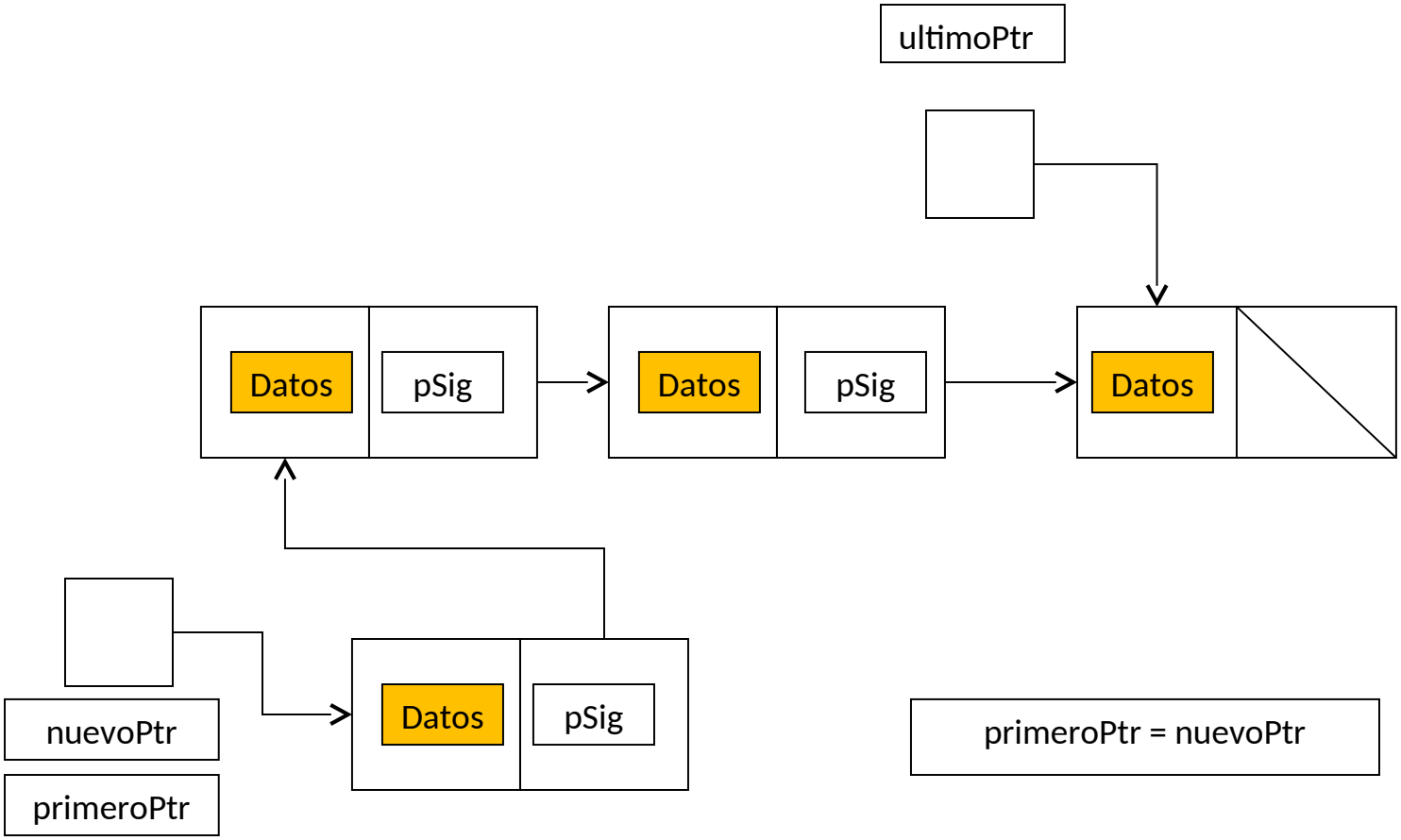
Operación: Insertar al frente

Paso 2: Enlazar nodo



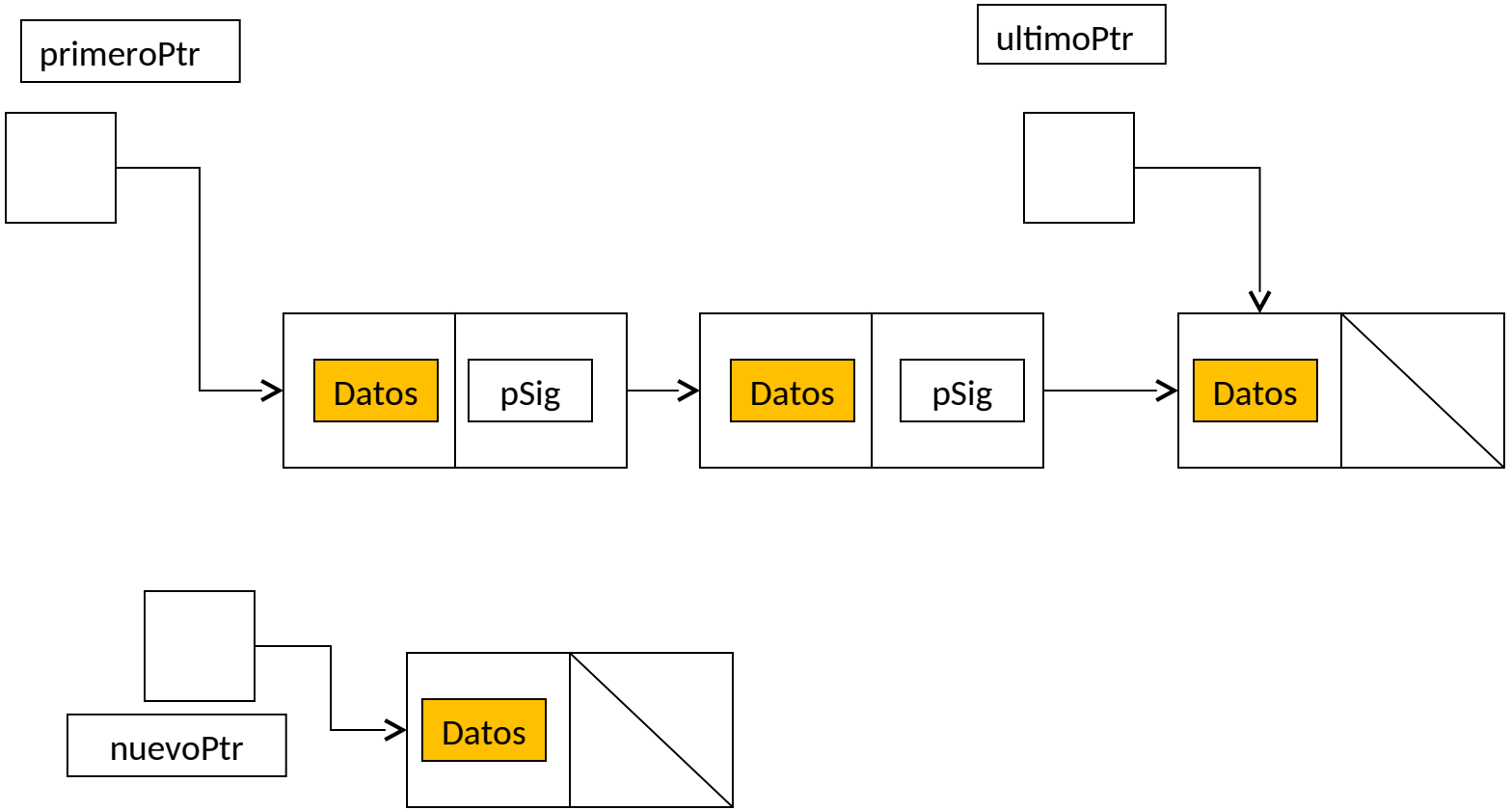
Operación: Insertar al frente

Paso 3: Redefinir comienzo de lista



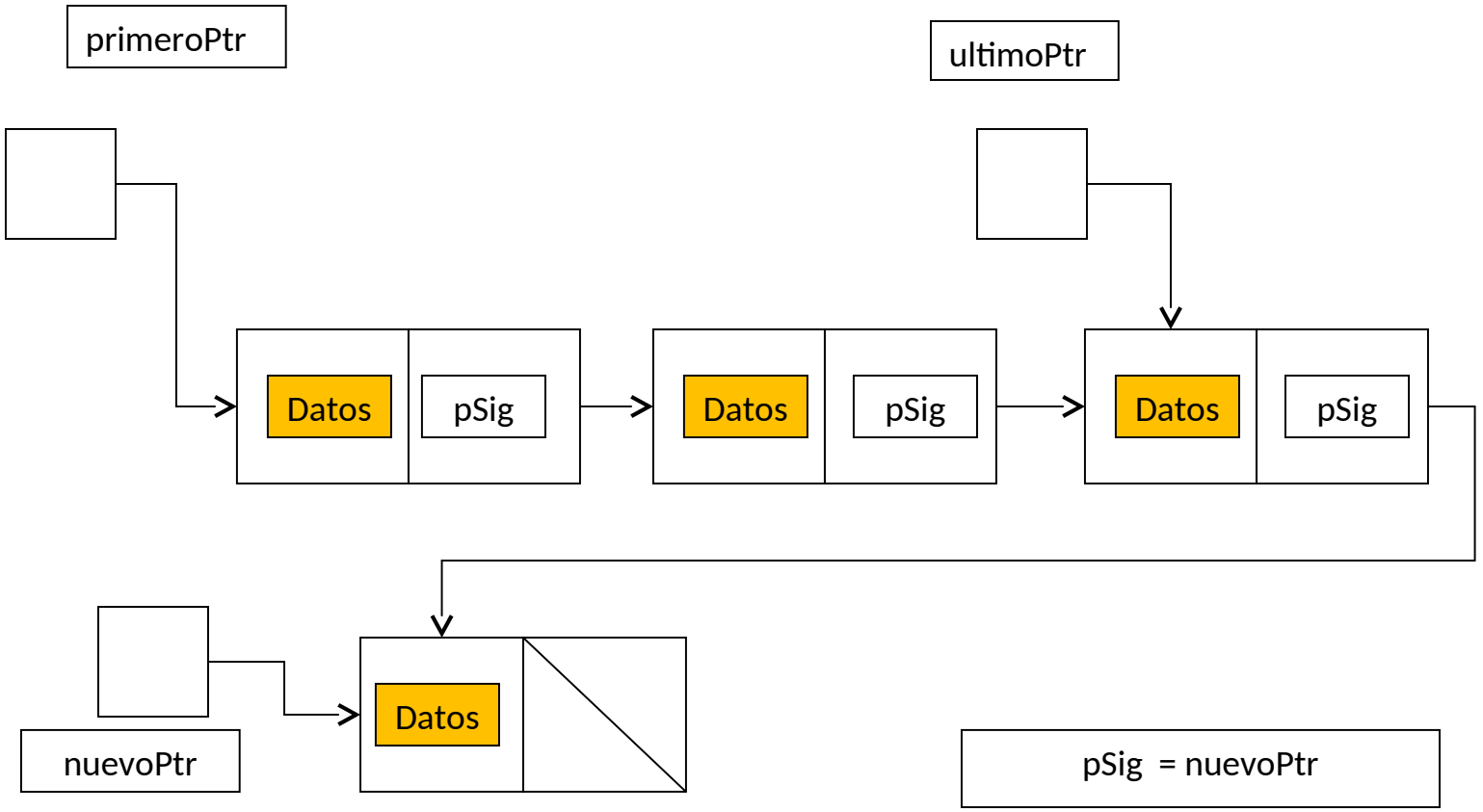
Operación: Insertar al final

Paso 1: Crear nodo



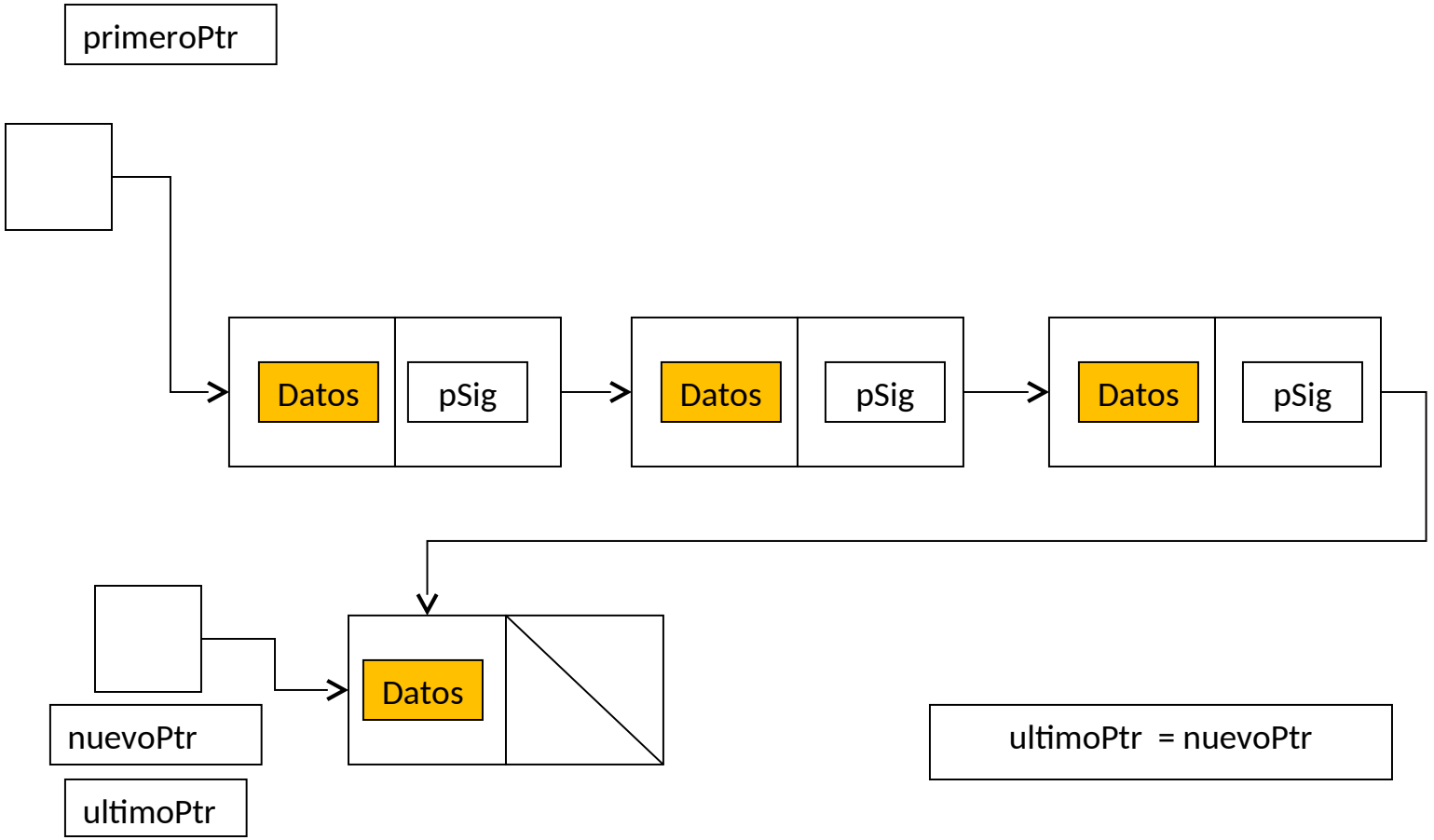
Operación: Insertar al final

Paso 2: Enlazar nodo



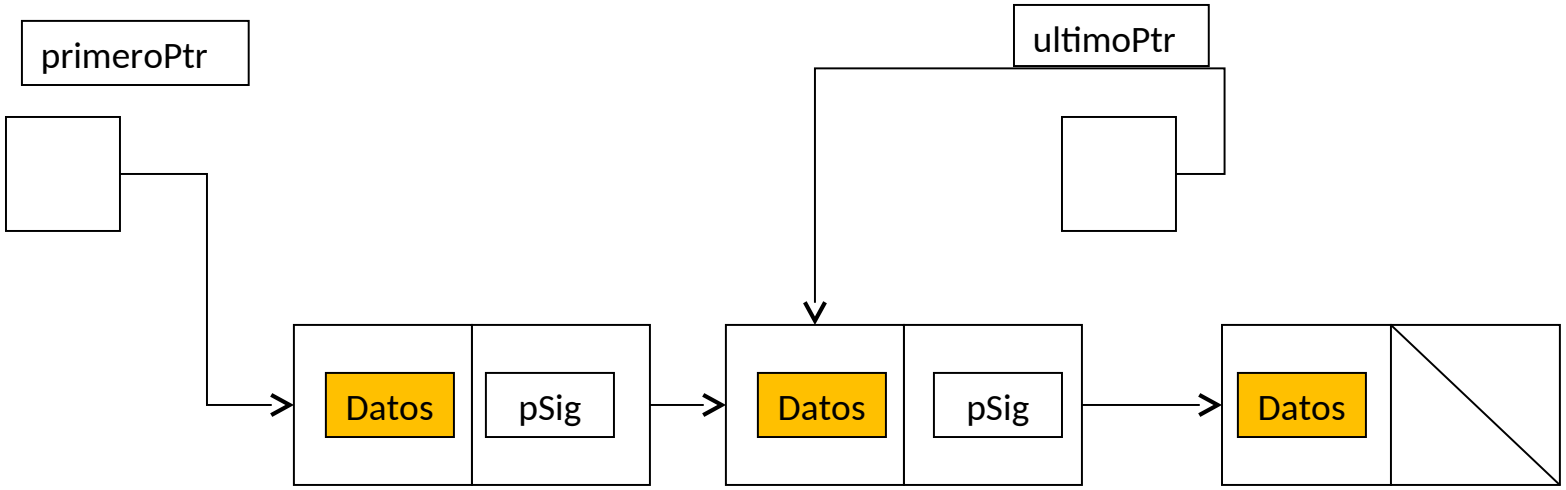
Operación: Insertar al final

Paso 3: Redefinir final de lista



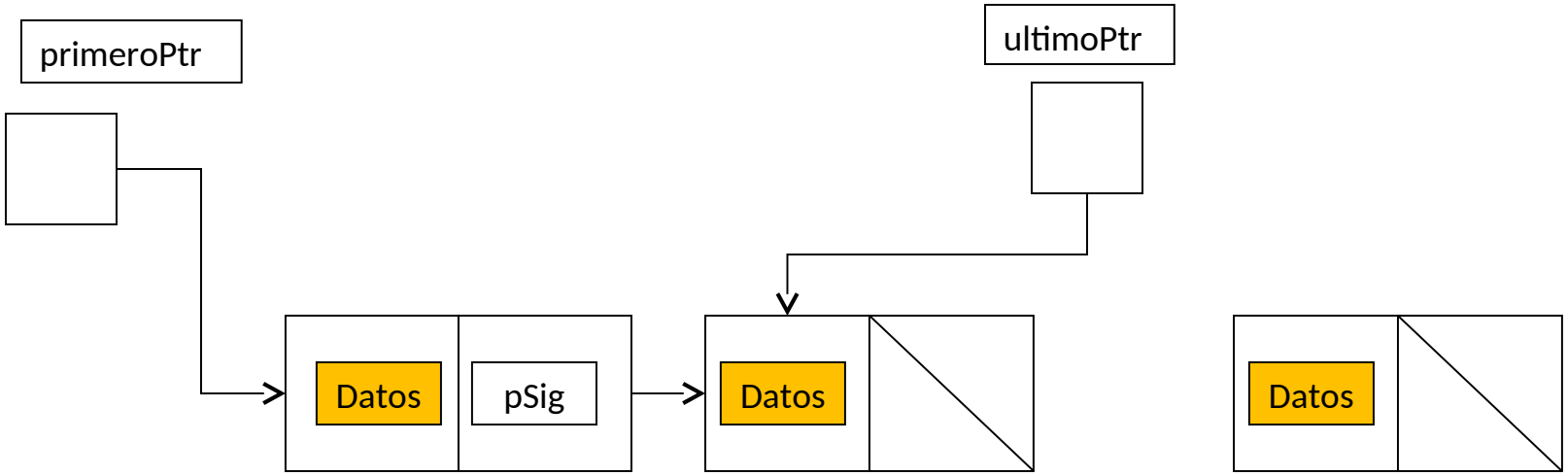
Operación: Eliminar del final

Paso 1: Orientar ultimoPtr



Operación: Eliminar del final

Paso 2: Borrar puntero a siguiente nodo



Código “artesanal” para crear lista simple

```
....

struct nodo
{
int CoordX;
int CoordY;
struct nodo *next;
};

int main()
{
struct nodo A,B,C,D,E,F, *temp;
A.CoordX = 1; A.CoordY = 2; B.CoordX = 3; B.CoordY = 4;
A.next = &B; B.next = &C; C.next = &D; D.next = &E; E.next = &F; F.next = NULL;

        clrscr();
        temp = &A;  printf("%d %d ", temp->CoordX, temp->CoordY);
        temp = &B;  printf("%d %d ", temp->CoordX, temp->CoordY);
        temp = &C;  printf("%d %d", (*temp).CoordX, (*temp).CoordY);
        temp = &D;  printf("%d %d", (*temp).CoordX, (*temp).CoordY);
        return 0;
}
```

Código sugerido para crear lista simple

```
....
struct nodo
{
int CoordX; int CoordY;
struct nodo *next;
}INICIO;

struct nodo *nuevonodo (struct nodo *);
struct nodo *nuevonodoFrente (struct nodo *);
void GrabarPunto (struct nodo *);

int main()
{
int i;
struct nodo *ULTIMO, *PRIMERO;
PRIMERO = &INICIO; PRIMERO->next = NULL;
ULTIMO = PRIMERO;

        for (i=0; i<10; i++)  // creamos  10 nodos
        {
                ULTIMO = nuevonodo(ULTIMO);
                GrabarPunto (ULTIMO);
        }
}
```

Funciones para crear lista simple

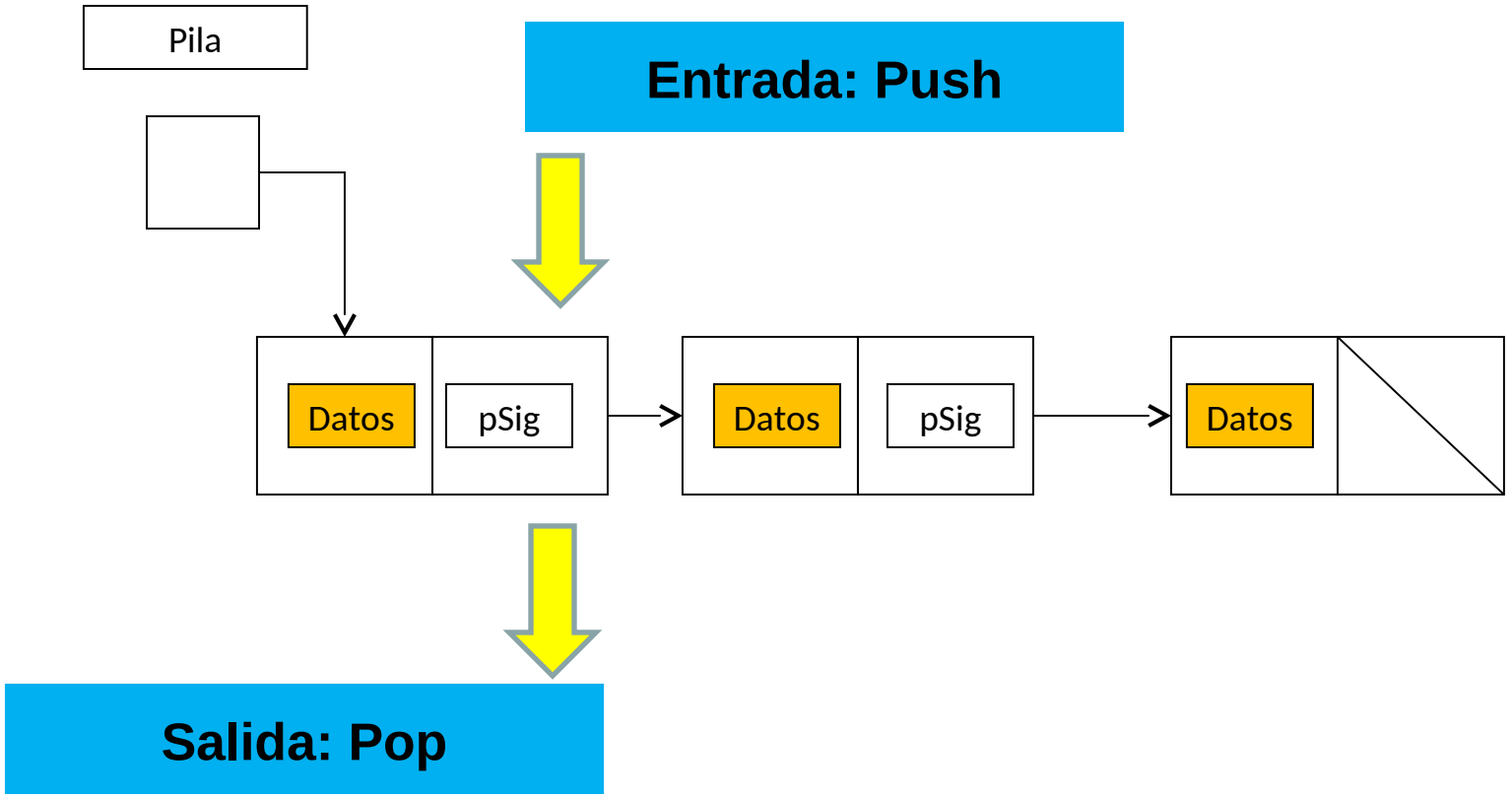
```
struct nodo *nuevonodo (struct nodo *actual)
{
struct nodo *nuevo;
nuevo = (struct nodo *) malloc (sizeof (struct nodo));
actual->next = nuevo;
nuevo->next = NULL;
return nuevo;
}

struct nodo *nuevonodoFrente (struct nodo *actual)
{
struct nodo *nuevo;
nuevo = (struct nodo *) malloc (sizeof (struct nodo));
nuevo ->next = actual;
return nuevo;
}

Void GrabarPunto (struct nodo *actual)
{
actual ->CoordX = serie.Latitud();
actual ->CoordY = serir.Longitu();
}
```

Pilas

Pila: Lista enlazada con modalidad LIFO

- **Lista Enlazada con modalidad LIFO (Last In First Out). Lo ultimo en entrar es lo primero en salir.**
- **Las operaciones de adición/eliminación se realizan por el comienzo de la Lista (Pila). Estas operaciones se conocen con el nombre de “push” y “pop” respectivamente.**

Pila: Lista enlazada con modalidad LIFO



Ejemplo Pila: Lista enlazada tipo LIFO

La pila de la computadora (Stack): A medida que un programa va llamando a sus funciones, existe un registro llamado Pila, que contiene la información de ubicación del proceso que ha llamado al proceso actual. Se trata de una estructura de tipo LIFO. La ultima función que ha sido invocada, es a donde hay que regresar primero, eliminándola de la lista.

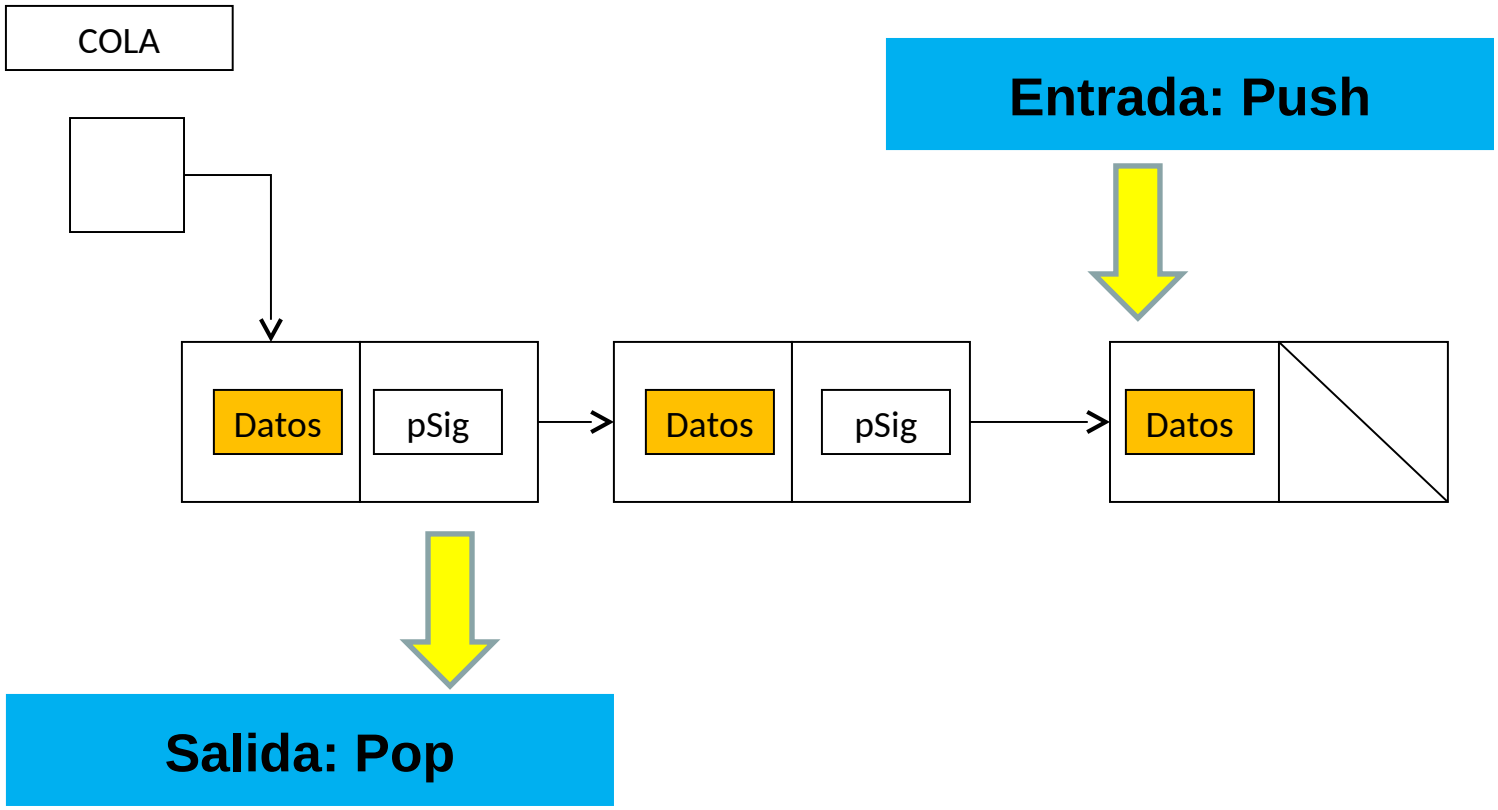
Stack overflow: Los desbordamientos de pila ocurren comúnmente debido a aplicaciones con muchos niveles de anidamiento, o en bucles infinitos donde hay llamadas a funciones internamente. Cuando se pierde el control sobre los lazos de control anidados, suele desbordarse la pila.

Colas

Cola: Lista enlazada con modalidad FIFO

- **Lista Enlazada con modalidad FIFO (First In First Out). El primero en entrar es el primero en salir.**
- **Las operaciones de adición se realizan por el final de la fila (Cola), operación push.**
- **Las operaciones de eliminación se realizan por el comienzo de la Cola, operación pop.**
- **Lista de Espera: de tareas, personas, procesos.**

Cola: Lista enlazada con modalidad FIFO



Cola: Lista enlazada con modalidad FIFO

Ejemplo: Un sistema de control puede dar tratamiento a las tareas solicitadas, a partir de ciertos eventos acorde a una estructura de tipo FIFO, dando prioridad según el momento en el que se produce la solicitud.

Una COLA permitiría dar tratamiento a determinadas tareas, a partir de la activación de ciertos sensores. Podemos decir:
COLA_Entradas
COLA_Tareas

Cola (FIFO): Funciones para agregar/quitar

```
struct nodo *agregar (struct nodo *fin, struct nodo *nuevo)
{
if ( fin )  {
fin ->next=nuevo;
fin = fin->next;
}
else fin = nuevo;
return fin;
}

struct nodo *quitar (struct nodo *inicio)
{
struct nodo *siguiente;
if ( inicio )  {
siguiente = inicio->next;
free(inicio);
}
else return NULL;
}
return siguiente;
}
```