DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Análisis de Algoritmos y Estructura de Datos

TDA lista enlazada

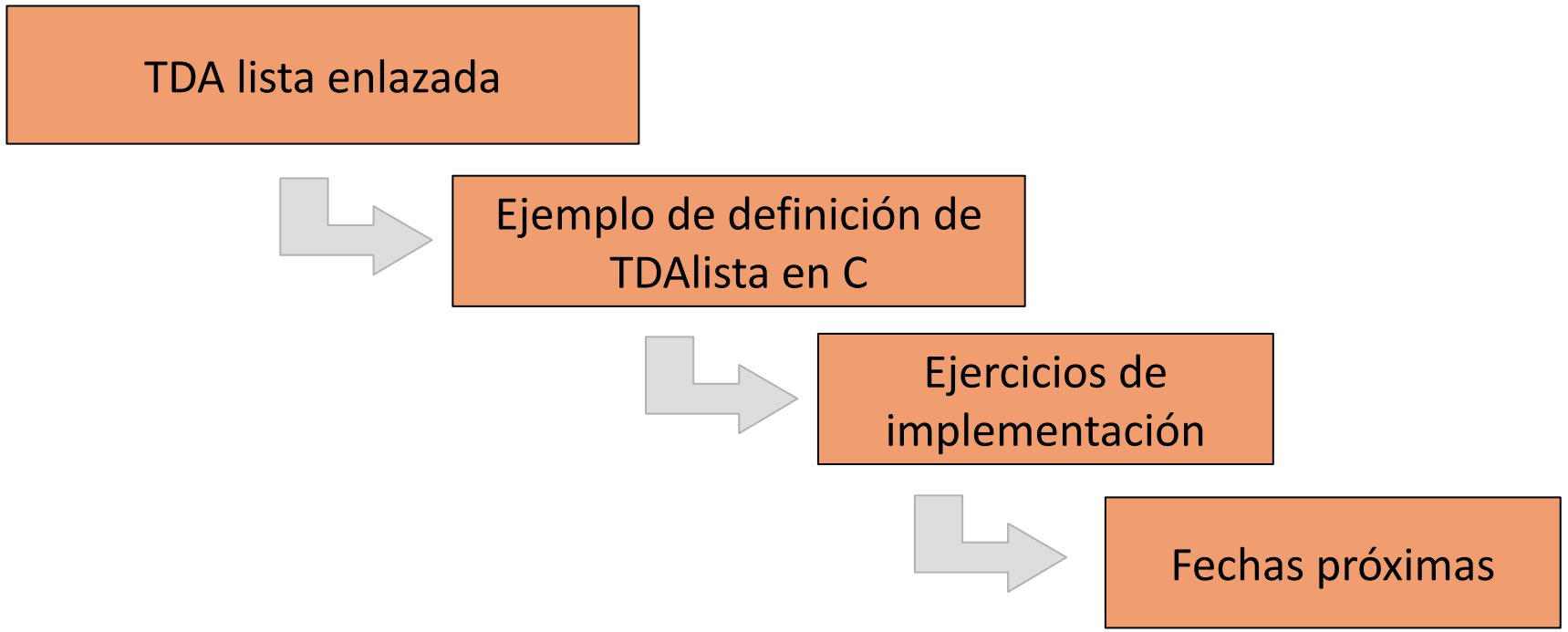
Prof. Violeta Chang C

Semestre 2 – 2023

TDA Lista Enlazada

- **Contenidos**:
  - Estructura de datos de TDA lista enlazada
  - Operaciones de TDA lista enlazada
- **Objetivos**:
  - Implementar TDA lista enlazada

Ruta de trabajo



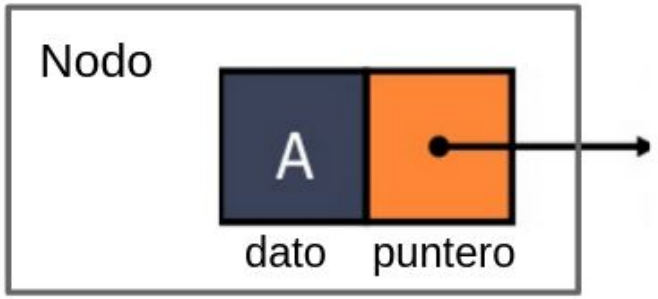
DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

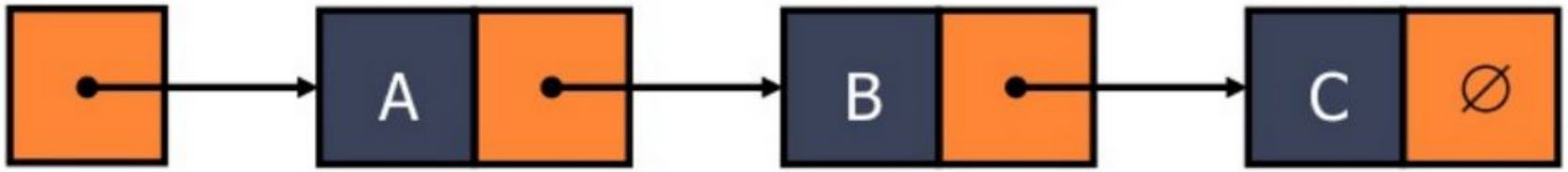
Especificación e implementación de TDA lista enlazada

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Especificación de TDA lista enlazada

- **Estructura de datos**:
  - Una lista enlazada (LE) es una secuencia de nodos conectados
  - A una lista con 0 nodos se le conoce como lista vacía
  - Cada nodo contiene:
    - Una parte de datos (cualquier tipo)
    - Un puntero al siguiente nodo de la lista
  - Cabeza: puntero al primer nodo
  - El último nodo apunta a nulo



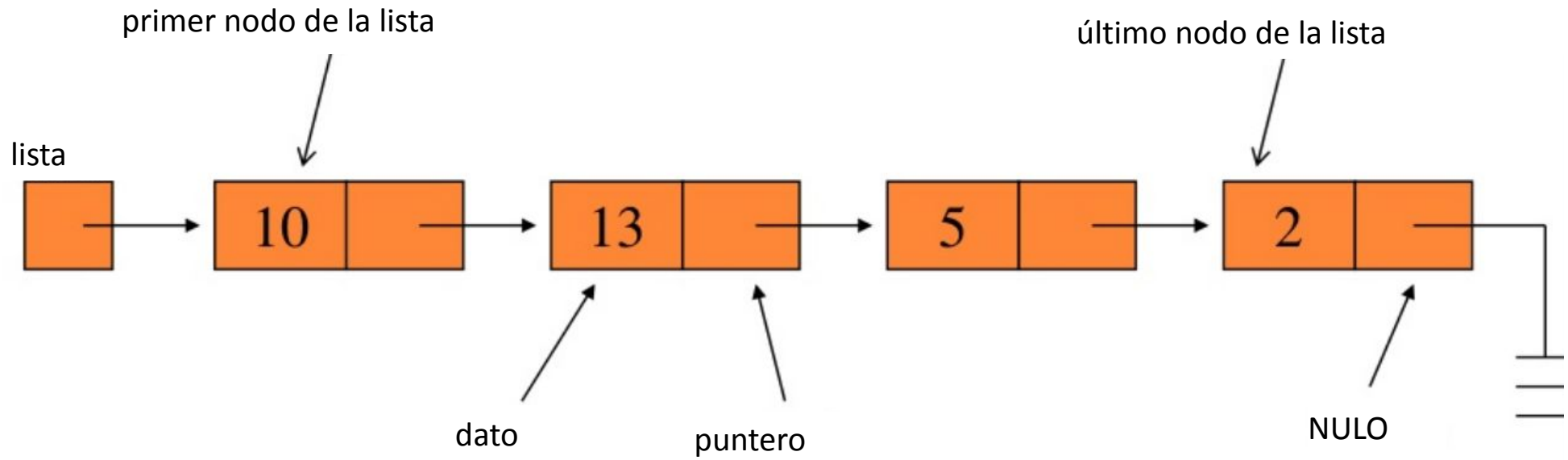


Khuram Shahzad, Data structures and algorithms

Especificación de TDA lista enlazada

- **Operaciones:**
  - **esListaVacía(L)**: determina si lista L está vacía o no
  - **insertarNodo(L,dato)**: inserta nodo con *dato* en lista L
  - **eliminarNodo(L,dato)**: elimina nodo con *dato* de lista L
  - **buscarNodo(L,dato)**: busca nodo con *dato* en lista L
  - **recorrerLista(L)**: muestra contenido de cada nodo de lista L

Especificación de TDA lista enlazada



Lista enlazada de enteros

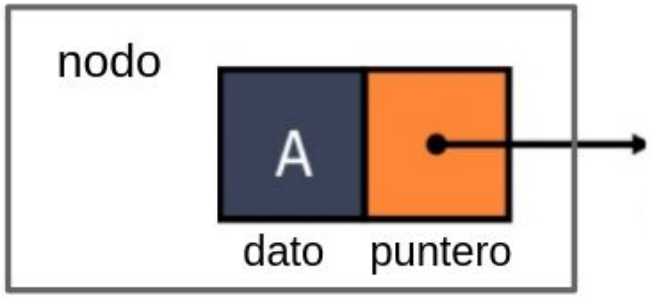
Khuram Shahzad, Data structures and algorithms

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Implementación de estructura de datos de TDA lista enlazada

- La estructura de datos que representa un nodo de una lista enlazada simple es la siguiente:

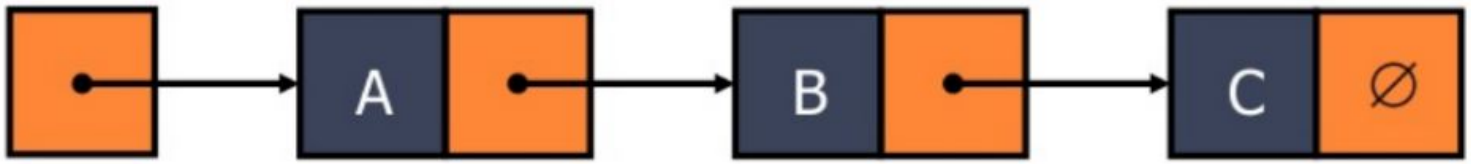
```
typedef struct nodoGenerico
{
  int dato;
  struct nodoGenerico* puntero;
}nodo;
```



Implementación de estructura de datos de TDA lista enlazada

- La estructura de datos que representa una lista es la siguiente:

```
typedef nodo* TDAlista;
```



Implementación de operaciones de TDA lista enlazada

```
TDAlista crearListaVacia()
{
  TDAlista lista=(TDAlista)malloc(sizeof(TDAlista));
  lista=NULL;
  return lista;
}

int esListaVacia(TDAlista lista)
{
  if (lista == NULL)
    return 1;
  else
    return 0;
}
```

Implementación de operaciones de TDA lista enlazada

```
void insertarInicio(TDAlista* lista, int dato)
{
  nodo* nuevo=(nodo*)malloc(sizeof(nodo));
  nuevo->dato=dato;
  nuevo->puntero=*lista;
  *lista=nuevo;
}

void eliminarInicio(TDAlista* lista)
{
  nodo* auxiliar;
  if(!esListaVacia(*lista))
  {
    auxiliar=*lista;
    *lista=(*lista)->puntero;
    free(auxiliar);
  }
}
```

Implementación de operaciones de TDA lista enlazada

```
void recorrerLista(TDAlista lista)
{
  if (!esListaVacia(lista))
  {
    nodo* auxiliar=lista;
    while (auxiliar!=NULL)
    {
      printf("%d ",auxiliar->dato);
      auxiliar=auxiliar->puntero;
    }
    printf("\n");
  }
  else
    printf("La lista está vacía\n");
}
```

Implementación de operaciones de TDA lista enlazada

- Usando la misma idea, se pueden implementar funciones básicas para trabajar con una lista enlazada simple:
  - **void insertarNodoFinal(TDAlista* lista, int dato)**
  - **void insertarNodoDespues(TDAlista* lista, int dato, int datoAnterior)**
  - **void eliminarFinal(TDAlista* lista)**
  - **void eliminarNodoDato(TDAlista* lista, int dato)**
  - **int obtenerNumeroNodos(TDAlista lista)**
  - **int buscarDato(TDAlista lista, int dato)**
  - **nodo* obtenerNodo(TDAlista lista, int posición)**
  - **void liberarLista(TDAlista* lista)**

Actividades de implementación

Actividad 1 - individual

1. Compilar y ejecutar **lab06-listaSimple.c**
2. Experimentar con las funciones implementadas en **TDAlista.h** haciendo llamadas desde función *main()* en **lab06-listaSimple.c**:
    - Insertar al inicio nodos en el siguiente orden: 4, 1, 3, 2
    - Recorrer la lista resultante
    - Eliminar nodo al inicio
    - Recorrer la lista resultante
    - Insertar al inicio nodo con valor 2
    - Recorrer la lista resultante

Actividad 2 - en conjunto

1. Implementar la siguiente función en **TDAlista.h** que devuelve el número de elementos de *lista*. La función debe devolver 0 en caso que *lista* esté vacía.

```
int obtenerNumNodos(TDAlista lista)
```

2. Evaluar las funciones creadas usando la lista de Actividad 1, generando secuencia de llamadas desde función *main()* en lab06-listaSimple.c para mostrar cuántos elementos tiene la lista

Actividad 3 - individual

1. Implementar la siguiente función en **TDAlista.h** que devuelve 1 en caso de encontrar *dato* en *lista*, y 0 en caso contrario:

```
int buscarDatoLista(TDAlista lista, int dato)
```

1. Evaluar las funciones creadas usando la lista de Actividad 1, generando secuencia de llamadas desde función *main()* en lab06-listaSimple.c para
    - buscar el dato 7
    - buscar el dato 1

Actividad 4 - individual

1. Usando **TDAlista.h** y **lab05-listaSimple.c,** implementar las siguientes funciones:

```
void insertarNodoFinal(TDAlista* lista, int dato);
void insertarNodoDespues(TDAlista* lista, int dato, int datoAnterior);
```

2. Evaluar todas las funciones creadas, generando secuencia de llamadas desde función *main()* en **lab05-listaSimple.c**

Actividad 5 - individual

1. Usando **TDAlista.h** y **lab05-listaSimple.c,** implementar las siguientes funciones:

```
void eliminarFinal(TDAlista* lista);
void eliminarDato(TDAlista* lista, int dato);
```

2. Evaluar todas las funciones creadas, generando secuencia de llamadas desde función *main()* en **lab05-listaSimple.c**

Actividad 6 - individual

1. Usando **TDAlista.h** y **lab05-listaSimple.c,** implementar la siguiente función:

```
nodo* obtenerNodo(TDAlista lista, int posicion);
```

2. Evaluar la función creada, generando secuencia de llamadas desde función *main()* en **lab05-listaSimple.c**

Entrega de actividad de laboratorio

- Entrega *obligatoria*
- Subir SOLO actividades 2, 3 y 4 de esta sesión en buzón de uVirtual, en único archivo **s5_apellido_nombre.zip**
- Se espera lab05-listaSimple.c y TDAlista.h (ambos modificados para responder a actividades 2, 3 y 4) comprimidos en archivo .zip
- Plazo: **hoy** dentro del horario de laboratorio de cada coordinación

Actividad de cierre

- Ir a menti.com e ingresar código 6290 5706

Próximas fechas...

- Resumen de la semana:
  - TDA lista enlazada
  - TDA lista enlazada circular
  - TDA lista doblemente enlazada

  ~~cátedra~~ – refuerzo1 – refuerzo2 – ~~lab1~~ – lab2

- Subsiguiente semana:
  - TDA pila
  - TDA cola

U2 - S5