* Repasando:
* Cuando se definen las variables referencia de un objeto, en un primer momento no se le asigna espacio de almacenamiento.
* Una vez que se usa el operador “New”, que es cuando se crea el objeto, de algún tipo, asignándole a alguna de las variables antes definidas, es ahí cuando se le asigna un espacio de almacenamiento.
* Podemos copiar la dirección que se encuentra en una referencia, que señala a un objeto, en otra referencia del mismo tipo, que fue definida anteriormente.
* Los **objetos** se definen en memoria dinámica, mientras que las **referencias** en memoria estáticas.
* Las referencias *señalan* al objeto, por lo que podemos definirlos como *enlaces* a los objetos.
* Estructura de datos:
  + Podemos clasificarlos por su:
* **Naturaleza**:
  + - Homogénea: Todos sus elementos corresponden a un mismo tipo de dato (vectores, tablas, matrices).
    - Heterogénea: En cambio, están compuesta por elementos de diferentes tipos (registros).
* **Forma de almacenamiento**:
  + - Memoria central:

Estática ⟶ Tamaño de almacenamiento conocido a priori (arreglos).

Dinámica ⟶ El tamaño de almacenamiento, en cambio, varía durante la ejecución del programa (listas, grafos, árboles)

* + - Memoria externa:

Archivos. Bases de datos.

* + Puntero: El concepto de puntero se aplica en la construcción de las estructuras dinámicas.
    - Es una variable estática, cuyo contenido es la dirección de una región en la memoria dinámica, donde esta región contiene información.
    - Mediante este puntero o referencia se puede acceder al contenido del objeto, es decir, a la información.
    - Un puntero se refiere a **una** cierta información, objeto. Pero este puede estar referido por varias referencias o punteros.
    - Además de acceder al contenido del nodo, también puede reservar y liberar espacio
  + Las estructuras de datos dinámicas basan su implementación en la creación de **nodos**.
  + Un nodo posee un campo destinado al almacenamiento de la información, y puede poseer uno o varios campos de tipo referencia que permiten señalar a otro u otros elementos que también son nodos y forman parte de la estructura de datos dinámica.
  + Gráficamente podemos representar a un nodo:

|  |  |  |
| --- | --- | --- |
| **NODO** | **Información** | **Enlace o puntero** |

* De esta manera el nodo se divide en dos partes:
  + Información: Contiene la información del elemento, donde esta puede ser de distintos tipos de datos, tanto primitivo como compuesto.
  + Puntero: Contiene la dirección del siguiente nodo de la lista. Es un dato de tipo referencia y nos va a permitir “señalar” a otra estructura idéntica (nodo).
* Cuando el campo referencia, del objeto nodo, no está asignado a un valor por defecto posee el valor “null”, es decir que no “señala” a alguna parte
* Generalmente en el “constructor” del nodo, dentro de la clase nodo, le asignaremos un valor inicial que le vamos a indicar (como parámetro), que se va a almacenar en el campo “información”, y en el campo enlace o puntero, el valor “null”, hasta que pueda ser agregado a la ED (estructura de datos dinámica).
* Podemos definir esta referencia del tipo nodo siguiendo dos criterios:
  + Sin asignarle espacio de almacenamiento;
  + Asignándole espacio de almacenamiento utilizando el operador “new”.
* LISTA:
  + Es una colección lineal de elementos llamados **NODOS** donde el orden de estos se establece mediante punteros o referencias.
  + Existe un puntero especial: *inicio* o *list*. Este nos permite acceder al primer elemento de la lista y a partir de esta poder acceder a los demás elementos.
  + Gráficamente:

![Gráfico, Gráfico de cajas y bigotes

Descripción generada automáticamente]()

* En una primera creación de la EDD lista, tiene la siguiente forma:

![Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente]()

Donde su primer valor va a ser “null” ya que no almacena algún elemento, es decir la que la lista estará vacía.

* En diagramación UML la EDD lista y la estructura nodo se representan:

![Diagrama

Descripción generada automáticamente]()