Imagen que contiene Icono

Descripción generada automáticamente

***Materia:***

***Estructura de dato I***

***Tarea Inicial***

***Integrante:***

***Thalia Delgado***

***Cedula:***

***K552709***

1. Buscar las diferencias entre Arreglos y las Matrices (Con su ejemplo)

Las matrices y arreglos (arrays) pueden ser descritas como **vectores multidimensionales.** Al igual que un vector, únicamente pueden contener datos de un sólo tipo, pero además de largo, tienen más dimensiones. En un sentido estricto, las matrices son una caso especial de un array, que se distingue por tener **específicamente dos dimensiones**, un “largo”" y un “alto”. Las matrices son, por lo tanto, una estructura con forma rectangular, con renglones y columnas. Los arreglos, por su parte, pueden tener un número arbitrario de dimensiones. Pueden ser cubos, hipercubos y otras formas. Su uso no es muy común en R, aunque a veces es deseable contar con objetos n-dimensionales para manipular datos. Como los arreglos tienen la restricción de que todos sus datos deben ser del mismo tipo, no importando en cuántas dimensiones se encuentren, esto limita sus usos prácticos.

Ejemplos de arreglos:

Un arreglo es un objeto contenedor que contiene un numero fijo de valores de un solo tipo. La longitud se establece cuando se crear el arreglo. Después de la creación, su longitud es fija.

Tabla

Descripción generada automáticamente

int numeros[10]; //Array de tipo entero de 10 de longitud

numeros[0] = 62;

numeros[1] = 12;

..

...

....

numeros[9] = 51;

Ejemplos de matrices:

Este tipo de arreglos son conocidos como matrices y pueden almacenar muchos más datos que los arreglos unidimensionales, los bidimensionales se componen de n filas por m columnas.

Gráfico

Descripción generada automáticamente

int numeros[2][3] = {{1,2,3},{3,4,5}}; //Matriz de tipo entero de 2 x 3

//Representación de matriz 10 x 10

1 2 3

3 4 5

1. Las diferencias y dónde aplicar una pila o una cola (con su ejemplo)

Diferencias entre una pila y una cola:

1. La pila sigue el mecanismo LIFO. La cola sigue el mecanismo FIFO para agregar y eliminar elementos.
2. En una pila, el mismo extremo se utiliza para insertar y eliminar los elementos. Por el contrario, se utilizan dos extremos diferentes en la cola para insertar y eliminar los elementos.
3. Como la pila solo tiene un extremo abierto, es la razón por la que se usa un solo puntero para referirse a la parte superior de la pila. Pero la cola utiliza dos punteros para referirse al frente y al final de la cola.
4. Stack realiza dos operaciones conocidas como push y pop, mientras que en la cola se conoce como enqueue y dequeue.
5. La implementación de la pila es más fácil, mientras que la implementación de la cola es difícil.
6. La cola tiene variantes como la cola circular, la cola de prioridad, la cola de doble finalización, etc. Por el contrario, la pila no tiene variantes.

Ejemplo de una pila:

Una pila es una estructura de datos lineal no primitiva. Es una lista ordenada donde se agrega el nuevo elemento y el elemento existente se elimina de un solo extremo, llamado como la parte superior de la pila (TOS). Como toda la eliminación e inserción en una pila se realiza desde la parte superior de la pila, el último elemento agregado será el primero en eliminarse de la pila. Esa es la razón por la que la pila se denomina tipo de lista Último en primer lugar (LIFO).

Teniendo en cuenta que el elemento al que se accede a menudo en la pila es el elemento superior, mientras que el último elemento disponible se encuentra en la parte inferior de la pila.

Diagrama

Descripción generada automáticamente

Ejemplo de una cola:

Una cola es una estructura de datos lineal que viene en la categoría del tipo no primitivo. Es una colección de tipo similar de elementos. La adición de nuevos elementos se lleva a cabo en un extremo llamado extremo posterior. De manera similar, la eliminación de los elementos existentes tiene lugar en el otro extremo, llamado Front-end, y es lógicamente un tipo de lista de Primero en entrar, primero en salir (FIFO).

Diagrama

Descripción generada automáticamente

1. Buscar: que se necesita para realizar una estructura de lista.

Una lista es una estructura dinámica de datos que contiene una **colección de elementos homogéneos (del mismo tipo) de manera que se establece entre ellos un orden**. Es decir, cada elemento, menos el primero, tiene un predecesor, y cada elemento, menos el último, tiene un sucesor.

Tanto las estructuras vistas en la sección anterior (pilas) como las que veremos en la siguiente (colas) son tipos de listas.

Podemos distinguir, **atendiendo a la organización de los nodos**, entre:

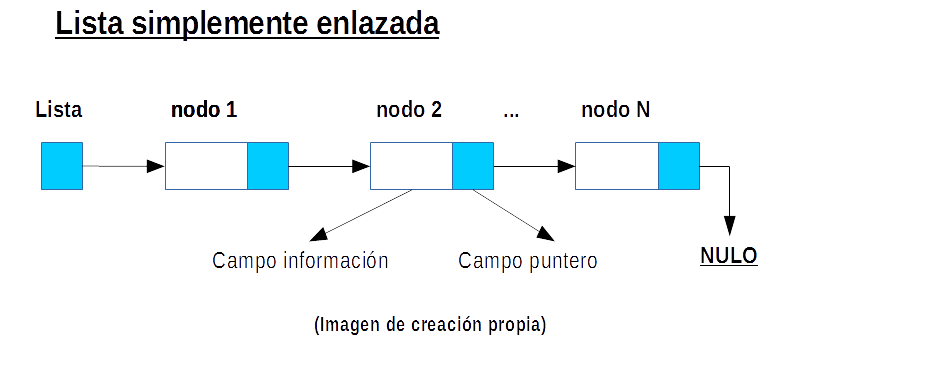
* **Listas simplemente enlazadas**: cada nodo tiene un campo que apunta al siguiente nodo.
* **Listas doblemente enlazadas**: cada nodo dispone de un puntero que apunta al siguiente nodo, y otro que apunta al nodo anterior.

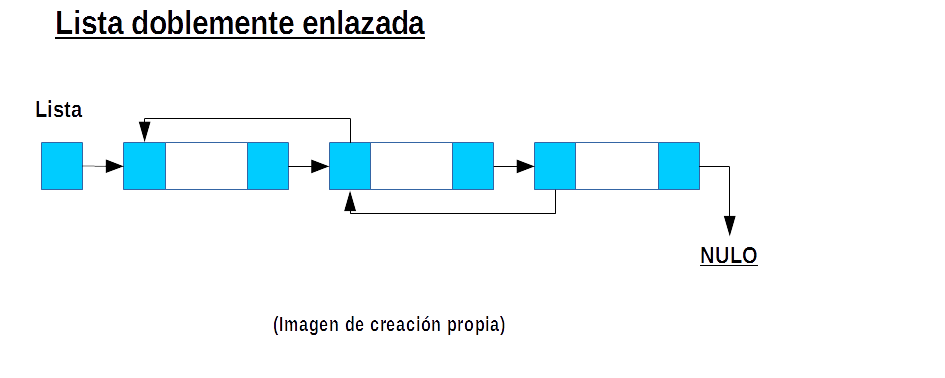
**Otra distinción** puede ser:

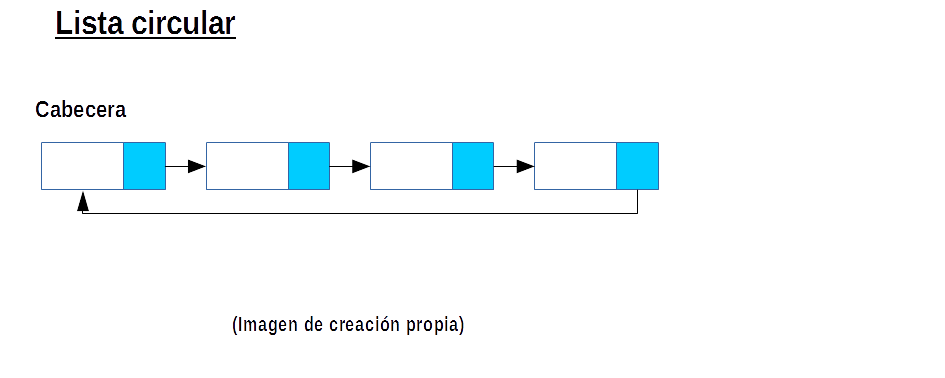
* **Listas lineales**: son listas que tienen un comienzo y un final.
* **Listas circulares**: en estas listas el último elemento apunta al primero, por lo tanto podríamos estar recorriéndolas siempre, ya que no tienen final.

.

**B) Representación gráfica**







**C) Operaciones básicas sobre una lista**

Las operaciones básicas que podemos realizar en cualquier lista, independiente del tipo que sea, son la siguientes:

* **Crear**: con esta operación se genera todo lo necesario para trabajar con una lista.
* **Insertar**: permite añadir un elemento a la lista. En este caso debemos indicar al programa si vamos a añadir el elemento nuevo al comienzo de la lista o al final de la misma.
* **Eliminar**: se usará para borrar un elemento de la lista. También podremos indicar si queremos borrar el primero o el último.
* **Vacía**: devolverá cierto si la lista está vacía.

Se podrán realizar otras operaciones como destruir la lista completamente, contar el número de elementos de la misma, añadir un elemento en una determinada posición, borrar un elemento de una determinada posición, comprobar si hay un dato en la lista.