ITEDES

Educación Digital

Módulo:

Fundamentos de Ingeniería de Software

Segmento:

Algoritmos y Estructuras de Datos

Tema:

Colecciones: Vectores y Listas

Prof. Germán C. Basisty
german.basisty@itedes.com

Índice de Contenidos

Vectores

Los **vectores**, también *llamados arreglos, arrays* o *matrices* son colecciones de variables que comparten nombre y tipo de dato, que están dispuestas de forma adyacente en la memoria, y que desde el punto de vista de la lógica de la aplicación están relacionadas de alguna manera.

A diferencia de las variables, los vectores tiene una determinada cantidad de dimensiones, y por cada dimensión tienen un índice y una magnitud. La magnitud es la longitud de la dimensión, y el índice es un número que identifica el elemento dentro de la dimensión.

Según el lenguaje cambia la sintaxis tanto de la declaración como de la nomenclatura de los vectores, por ejemplo, en *BASH* los vectores no se declaran (se usan directamente ya que *BASH* es un lenguaje de tipado débil) y las dimensiones / índices se denotan con los símbolos "[" y "]"

Los vectores definen su tamaño en el momento de ser declarados .

Ejemplos

- Vector de una dimensión de tipo entero: numero[5]

numero

Posición [0]	Posición [1]	Posición [2]	Posición [3]	Posición [4]

Para asignarle un valor nodo de un vector, llamarlo por su nombre e índice, por ejemplo:

```
numero[1] = 16
numero[3] = 12
numero[0] = 11
```

- Vector de dos dimensiones de tipo entero: numero[4][3]

numero

Posición [0][0]	Posición [0][1]	Posición [0][2]	Posición [0][3]
Posición [1][0]	Posición [1][1]	Posición [1][2]	Posición [1][3]
Posición [2][0]	Posición [2][1]	Posición [2][2]	Posición [2][3]

Para asignarle un valor nodo de un vector, llamarlo por su nombre e índice, por ejemplo:

```
numero[1][0] = 16
numero[3][3] = 12
numero[0][0] = 11
```

Listas

Las listas son estructuras de datos mas modernas que los vectores, pero comparten la definición de ser una colección de datos del mismo tipo (esto no es del todo cierto, ya que por ejemplo en **Python** es posible crear listas de distintos tipos de elementos, o en **Java** se puede obtener el mismo efecto creando listas del tipo **Object**) relacionados entre si de alguna manera.

A diferencia de los vectores, con las listas es posible agregar, borrar y reemplazar elementos.

Python

```python
nombres = ['Mateo', 'Marcos']

nombres.append('Lucas')
nombres.append('Juan')

tamanio = len(nombres)

print('Tamaño de la lista: ' + str(tamanio))

# Imprimimos la lista utilizando una variable iteradora (nombre)
for nombre in nombres:
    print(nombre)

print()

# Imprimimos la lista en función de su posición
for indice in range(4):
    print(str(indice) + ": " + nombres[indice])

# Borramos un elemento en función del valor
nombres.remove('Mateo')

print()

# Borramos un elemento en función de su posición
del nombres[1]

print(nombres)
```

Comandos útiles:

- lista.append(elemento) para agregar un elemento a la lista
- len(lista) nos devuelve el tamaño de la lista (un entero)
- del lista[4] elimina de la lista el elemento que está en la posición 4
- lista.remove('Elemento') elimina de la lista el elemento "Elemento"

Java

```java
import java.util.ArrayList;

public class Listas {
    public static void main(String args[]) {
        // Creamos una lista tipo String. Recordar que Java es un
        // lenguaje de tipado fuerte
        ArrayList<String> nombres = new ArrayList<String>();

        nombres.add("Mateo");
        nombres.add("Marcos");
        nombres.add("Lucas");
        nombres.add("Juan");

        Integer tamanio = nombres.size();
        System.out.println("Tamanio de la lista: "
            + tamanio.toString());

        // Imprimimos la lista utilizando una variable iteradora
        // (nombre)
        for(String nombre : nombres)
            System.out.println(nombre);

        System.out.println();

        // Imprimimos la lista en funcion de su posicion
        for(Integer i = 0; i < 4; i++)
            System.out.println(nombres.get(i));

        // Borramos un elemento en funcion del valor
        nombres.remove("Mateo");

        // Borramos un elemento en funcion de su posicion
        nombres.remove(1);

        System.out.println();

        System.out.println(nombres);
    }
}
```

Comandos útiles:

- Las listas en java se implementan mediante la clase ArrayList. Importar según sea necesario.
- lista.size() devuelve el tamaño de la lista (entero)
- lista.add(Elemento) agrega un elemento a la lista
- lista.remove(indice) elimina un elemento según su posición
- lista.remove(Elemento) elimin un elemento de la lista según su valor

C#

```
using System;
using System.Collections.Generic;

namespace listas.net
{
    class Program
    {
        static void Main(string[] args)
        {
            List<string> nombres= new List<string>();

            nombres.Add("Mateo");
            nombres.Add("Marcos");
            nombres.Add("Lucas");
            nombres.Add("Juan");

            Console.WriteLine("Tamaño: " + nombres.Count);

            Console.WriteLine();

            // Imprimimos la lista utilizando una variable iteradora
            // (nombre)
            foreach(string nombre in nombres)
                Console.WriteLine(nombre);

            Console.WriteLine();

            // Imprimimos la lista en funcion de su posicion
            for(int i = 0; i < 4; i++)
                Console.WriteLine(nombres[i]);

            // Borramos un elemento en funcion del valor
            nombres.Remove("Mateo");

            // Borramos un elemento en funcion de su posicion
            nombres.RemoveAt(1);

            Console.WriteLine();

            foreach(string nombre in nombres)
                Console.WriteLine(nombre);
        }
    }
}
```

Comandos útiles:

- Las listas en C# se implementan mediante la clase List. Importar según sea necesario.
- lista.Count devuelve el tamaño de la lista (entero)
- lista.Add(Elemento) agrega un elemento a la lista
- lista.RemoveAt(indice) elimina un elemento según su posición
- lista.Remove(Elemento) elimin un elemento de la lista según su valor

JavaScript

```
let nombres = ['Mateo', 'Marcos'];

nombres.push('Lucas');
nombres.push('Juan');

console.log(nombres.length);

// Imprimimos la lista utilizando una variable iteradora
// (nombre)
nombres.forEach(nombre => {
    alert(nombre);
});

// Imprimimos la lista en funcion de su posicion
for(let i = 0; i < 4; i++)
    alert(nombres[i]);

// Borramos un elemento en funcion de su posicion
nombres.splice(1, 1);

nombres.forEach(nombre => {
    alert(nombre);
});
```

Comandos útiles:

- Las listas en JavaScript se llaman arrays
- lista.length devuelve el tamaño de la lista (entero)
- lista.push(Elemento) agrega un elemento a la lista
- lista.splice(indice, cantidad) elimina elementos según su posición y cantidad

Ejercitación

1) Desarrollar un juego de ahorcado. Requerimientos:

- Se debe cargar una lista de palabras.
- Se debe sortear una palabra de la lista.
- Se permiten hasta 5 equivocaciones al adivinar las letras.

Presentar código fuente en Python, Java y C#.

2) Desarrollar un algoritmo que rellene un vector con los números impares comprendidos entre 1 y 100 y los muestre en pantalla en orden ascendente. Presentar diagrama de flujo, pseudocódigo y código fuente funcionando en BASH, Python, Java, C#, y JavaScript + HTML.

3) Desarrollar un algoritmo que lea 5 números por teclado y los guarde en un vector, los copie a otro vector multiplicados por 2 y muestre el segundo vector. Presentar diagrama de flujo, pseudocódigo y código fuente funcionando en BASH, Python, Java, C#, y JavaScript + HTML.