**Colecciones en Java**

El API de Java nos proporciona el framework de las colecciones, que nos permite utilizar diferentes estructuras de datos para almacenar y recuperar objetos de cualquier clase. Dichas colecciones no forman parte del lenguaje, sino que son clases definidas en el paquete  java.util

Para crear una colección usaremos la siguiente estructura:

Coleccion<Clase> nombre = **new** Coleccion<Clase>();

Donde Coleccion es una clase de este framework que queramos utilizar según la estructura de almacenamiento que nos interese y Clase representa el tipo de datos a almacenar.

Hay tres tipos de colecciones, cada uno con un interfaz común y diferentes implementaciones. Las diferentes implementaciones de un mismo interfaz realizan la misma tarea aunque la diferencia está en que unas implementaciones son más rápidas en algunas operaciones y más lentas en otras.

**Listas** – estructura secuencial, donde cada elemento tiene un índice o posición.

ArrayList

La clase ArrayList en Java, es una clase que permite almacenar datos en memoria de forma similar a los Arrays, con la ventaja de que el número de elementos que almacena, lo hace de forma dinámica, es decir, que no es necesario declarar su tamaño como pasa con los Arrays.

Ej:

ArrayList<tipo\_dato> nombre\_del\_arreglo= new ArrayList<tipo\_datos>();

ArrayList<Trabajador> traba=new ArrayList<Trabajador>();

Los ArrayList se utilizan en general con objetos, es decir que pueden guardarse objetos en cada posición del arreglo. A diferencia de un arreglo de objetos que es de un largo fijo, el ArrayList es dinámico, va creciendo a medida que se ingresan datos, y se va achicando a medida que se le quitan los datos.

Crearemos un proyecto con el nombre de “SolucionEjemploArrayListFutbol”

Craremos una clase llamada “Partido”

Con los atributos:

private String codigoPartido;

private String equipoLocal;

private String equipoVisitante;

private int golesLocal;

private int golesVisitante;

Sus respectivos get y set, mas su constructor, además heremos un método para ver que equipo gano el partido o si empataron.

public String ganador()

{

if(golesLocal>golesVisitante)

return "Ha ganado el equipo Local "+equipoLocal+ "Con "+golesLocal+" goles";

else

if(golesLocal<golesVisitante)

return "Ha ganado el equipo Visitante "+equipoVisitante+ "Con "+golesVisitante+" goles";

else

return "Los equipos han empatado "+golesLocal+" = "+golesVisitante;

}

Luego crearemos otra clase a la que nombraremos “Torneo”

Esta clase tendrá dos atributos, nombre del torneo y un ArrayList de tipo Partido, esta clase será una clase conocida como Manejadora o Repositorio.

Cuando declara el Arraylist java pide que se importe una librería

import java.util.ArrayList;

Esto va antes del nombre de la clase.

private String nombreTorneo;

private ArrayList<Partido> parti =new ArrayList<Partido>();

En esta clase haremos lo siguiente, además de los get, set y constructor, cuando creamos el constructor veremos que el ArrayList no aparece en la lista, porque ya está inicializado.

Ahora, buscaremos un partido especifico por el código de partido, generalmente se usa un campo que no se repita, que sea único, a no ser que se especifique otra cosa.

Para buscar usaremos un ciclo, el for en este caso, el método recibirá el código que deseo buscar dentro del arreglo y como es String debo usar el compareToIgnoreCase()

public int buscarCodigo(String codigo)

{

for (int i = 0; i < parti.size(); i++)

{

if(codigo.compareToIgnoreCase(parti.get(i).getCodigoPartido())==0)

{

return i;

}

}

return -1;

}

Si retorna el valor i, significa que encontró el código en el arreglo, si devuelve -1, significa que no se encontró el código

Ahora agregaremos un partido especifico al arreglo, antes de agregarlo verificaremos que ese partido no exista en el arreglo, esto se hace por el código.

public String ingresarPartido(Partido pp)

{

if(buscarCodigo(pp.getCodigoPartido())==-1)

{

parti.add(pp);

return "\npartido agregado correctamente";

}

else

return "\nPartido ya existe, no se puede ingresar";

}

Ahora eliminaremos un partido específico, también usando el código

public String eliminarPartido(String codigo)

{

int pos;

pos=buscarCodigo(codigo);

if(pos==-1)

return "Partido no existe... no se puede eliminar";

else

{

parti.remove(pos);

return "\nPartido eliminado ";

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Bien, con esto terminamos la clase Torneo. Ahora vamos a la aplicación:

int op,pos, golessL, gloesV;

String codigo,eqLocal,eqvisitante, nom;

Partido par=null;

System.out.print("\n\tIngrese nombre del Torneo: ");

nom=Leer.dato();

Torneo tor=new Torneo(nom);

do {

System.out.println("\n\n\tTorneo\n" +

"1. Ingresar Partido\n2. Mostrar los datos del Partido X\n"+

"3. Eliminar un partido \n4. Mostrar ganador ?n5. Salir");

System.out.print("\tElija opción: ");

op=Leer.datoInt();

switch (op) {

case 1:

System.out.print("Codigo del partido: ");

codigo=Leer.dato();

System.out.print("Nombre equipo Local: ");

eqLocal=Leer.dato();

System.out.print("Nombre equipo Visitante ");

eqvisitante=Leer.dato();

System.out.println("Cantidad goles equipo Local: ");

golessL=Leer.datoInt();

System.out.println("Cantidad goles Visitante ");

gloesV=Leer.datoInt();

par=new Partido(codigo, eqLocal, eqvisitante, golessL, golessL);

System.out.println(tor.ingresarPartido(par));

break;

case 2:

System.out.println("Ingrese codigo del partido a Buscar: ");

codigo=Leer.dato();

pos=tor.buscarCodigo(codigo);

if(pos==-1)

System.out.println("El partido que busca no existe, imposible mostrar sus datos");

else

{

System.out.println("Mostrando informacion del partido");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

System.out.println("Codigo: "+tor.getParti().get(pos).getCodigoPartido());

System.out.println("Equipo Local : "+tor.getParti().get(pos).getEquipoLocal());

System.out.println("Equipo Visitante: "+tor.getParti().get(pos).getEquipoVisitante());

System.out.println("Goles local: "+tor.getParti().get(pos).getGolesLocal());

System.out.println("Goles visitante : "+tor.getParti().get(pos).getGolesVisitante());

}

break;

case 3:

System.out.println("Ingrese codigo del partido a eliminar: ");

codigo=Leer.dato();

System.out.println(tor.eliminarPartido(codigo));

break;

case 4:

System.out.println("Ingrese codigo del partido a Buscar: ");

codigo=Leer.dato();

pos=tor.buscarCodigo(codigo);

if(pos==-1)

System.out.println("El partido no existe, imposible mostrar resultado final");

else

System.out.println("Nota Final: "+tor.getParti().get(pos).ganador());

break;

case 5:

System.out.println("Adios");

break;

default:

System.out.println("Debe ingresar una opcion correcta");

}

} while (op!=5);