Algoritmos y estructuras de datos

Laboratorio 14 de Mayo

Apuntes

-Siempre usar el new para crear un nuevo objeto.
-¿Que carajos es un hashmap?

Tipos parametricos Nos permiten definir interfaces genericas, se usa mayusculas para indicar que pueden tomar cualquier valor, no obstante si nuestra implementacion requiere de orden, entonces T debe ser *comparabl*

Recorriendo colecciones ¿Como lo hacemos? for, foreach, while y un indice, *¿pero donde estan definidas las formas en las que se recorren los objetos?*
-Los objetos son privados, y tienen que proveer una forma de recorrerlos.

Iteradores: Un iterador es una manera abstracta de recorrer colecciones, independientemente de su estructura.
-¿Esta posicionado sobre un elemento?
-Obtener el elemento actual.
-Avanzar al siguiente elemento.
- una que me falto jaja

Iteradores en java: Java nos provee una interface **Iterator<E>**, la cual nos obliga a crear dos funciones, una que nos indica si hay un siguiente elemento, y otra que nos devuelve el elemento en esa posicion. Para las distintas implementaciones que creemos tendremos que implementar nuestros propios iteradores usando esta interface.

Listas simplemente enlazadas Es una estructura que sirve para representar una secuencia de elementos, distin del vector.
Guardamos la informacion en nodos, cada nodo tiene un determinado dato y una referencia al siguiente elemento
¿una coleccion de duplas?
Asumiendo que tenemos una referencia al primer elemento, y una variable *size*. ¿Cual es su invariante de representacion?
-Si la lista esta vacia, entonces *primero* es *null*, y *size* es 0
-Si la lista no esta vacia, entonces *primero* apunta al primer elemento, y size es la cantidad de nodos.
-Todos apuntan al siguiente, menos el ultimo.
-El ultimo nodo apunta a null.
No definiremos formalmente el invariante de representacion porque la logica de primer orden no nos basta para describir esta clase de estructuras.
acá estamos mezclando un poco la funcion de abstraccion y el predicado de invariante

Principales ventajas Manejo mas fino de la memoria (ya no tenemos que reservar memoria por adelantado), agregar un elemento en cualquier punto de la estructura es muy eficiente, solo implica agregar un nodo y redirigir una referencia. Tambien es muy eficiente para reacomodar los elementos en otro orden.

Desventajas Acceder a los elementos no es tan simple como en un vector, hay que recorrer elemento a elemento hasta llegar al correcto, perdemos el acceso arbitrario a elementos.

```
package aed;
import java.utils.*;
class ListaDeInts{
   private primero;
   private class Nodo {
      private dato = new tuple[2]
```

```
    }
}
```