Contenedores

Secuenciales (elementos ordenados):

* List (lista dbl-enlazada)
* Vector (arreglo lineal)
* Deque (doble cola)

Asosiativos (claves)

* set(conjunto)
* map(correspondencia)

Adaptadores:

* stack (pila)
* queue (cola)
* piority\_queue (cola con prioridades)

otros:

* array

iteradores:

* de acceso aleatorio, secuenciales
* de entrada, de salida
* direccionales, bidireccionales
* de inserción, etc.

Map

* Guarda un conjunto de asociaciones entre objetos, pares clave-valor
* Clave y valor pueden ser de tipos diferentes
* No puede haber claves repetidas
* El tipo de clave debe ser ordenable (operador <)
* Insertar o buscar una clave en un map es rápido

El mapa se ordena de menor a mayor

**Crear un mapa:**

map<string, int> agenda; *//asocia ints a strings //string es la "clave", int es el valor*

**Guardar datos en el mapa:**

agenda["Fulano"] = 15456324; *//el indice es la clave(string)*

agenda["Mengueche"] = 15415124;

agenda["Chuck Norris"] = 1352524;

agenda["Juan Perez"] = 15456535;

**Consultar un dato del mapa, si sé que existe:**

cout << "El nro de Chuck es: " << agenda["Chuck Norris"] << endl;

*//si no está en la agenda se crea automaticamente*

**Consultar un dato cuando no se si existe:**

auto it = agenda.find("Wally"); *//si no lo encuentra retorna end()*

if(it == agenda.end())

cout << "No encontrado a Wally";

else

cout << "Nro de Wally: " << it->second; *//cada elemento del mapa es un struct con su clave(first) y su valor(second)*

for( auto it=agenda.begin(); it!=agenda.end(); ++it) {

//auto es map<string, int>::iterator

cout << it -> first << "esta asociado a " << it -> second << endl;

}

for(auto &p : m){

cout << p.first << "esta asociado a " << p.second << endl;

}

agenda.size() ; *// para saber la cantidad de3 valores*