1. Bibliotecas Estándar de C++ (STL)

La Biblioteca Estándar de Plantillas (STL) es una parte fundamental de la Biblioteca Estándar de C++, que proporciona una colección de clases y funciones genéricas para manipular ***estructuras de datos.***

Las principales componentes de la STL son:

Contenedores: Estructuras de datos como ***vector, list, set, map***, etc.
Algoritmos: Funciones genéricas como ***sort, find, accumulate***, entre otros.
Iteradores: Objetos que permiten recorrer los elementos de los contenedores.

2. Vectores

Los vectores (std::vector) son arreglos dinámicos que pueden cambiar de tamaño automáticamente.

Creación:

```
std::vector<int> vec; // Vector vacío
std::vector<int> vec(10); // Vector con 10 elementos inicializados a 0
std::vector<int> vec(10, 5); // Vector con 10 elementos inicializados a 5
```

Modificación:

Añadir elementos: push_back() añade un elemento al final.

```
vec.push_back(7);
```

Eliminar elementos: pop_back() elimina el último elemento.

```
vec.pop_back();
```

Acceso a elementos: at(), [] permiten acceder a elementos.

```
int value = vec.at(2);

int value = vec[2];
```

Redimensionar: resize() cambia el tamaño del vector.

```
vec.resize(5); // Cambia el tamaño a 5 elementos
```

Iteración:

Iteradores normales:

```
for(auto it = vec.begin(); it != vec.end(); ++it) {
     std::cout << *it << " ";
}
```

Rango de for:

```
for(int val : vec) {
     std::cout << val << " ";
}
```

3. Listas, Conjuntos, Mapas

Listas (std::list):

Uso: Doble enlace, permite inserciones/eliminaciones eficientes en cualquier parte.

Creación:

```
std::list<int> lst = {1, 2, 3, 4};
```

Inserción: push_back(), push_front()

```
lst.push_back(5);
lst.push_front(0);
```

Eliminación: pop_back(), pop_front()

```
lst.pop_back();
lst.pop_front();
```

Conjuntos (std::set):

Uso: Almacena elementos únicos en orden.

Creación:

```
std::set<int> mySet = {1, 2, 3, 4};
```

Inserción: insert()

```
mySet.insert(5);
```

Eliminación: erase()

```
mySet.erase(3);
```

Mapas (std::map):

Uso: Almacena pares clave-valor en orden.

Creación:

```
std::map<int, std::string> myMap;
```

Inserción: insert(), []

```
myMap[1] = "one";
myMap.insert({2, "two"});
```

Eliminación: erase()

```
myMap.erase(1);
```

4. Algoritmos

Los algoritmos en STL son funciones genéricas para operar sobre contenedores.

sort: Ordena los elementos de un contenedor.

```
std::sort(vec.begin(), vec.end());
```

find: Busca un valor en un rango.

```
auto it = std::find(vec.begin(), vec.end(), 3);
if (it != vec.end()) {
     std::cout << "Found: " << *it;
}
```

accumulate: Suma o combina valores en un rango.

```
int sum = std::accumulate(vec.begin(), vec.end(), 0);
```

5. Iteradores

Los iteradores son abstracciones que permiten recorrer los elementos de los contenedores.

Iteradores de contenedores:

```
std::vector<int>::iterator it;
for(it = vec.begin(); it != vec.end(); ++it) {
     std::cout << *it << " ";
}
```

Iteradores inversos:

```
std::vector<int>::reverse_iterator rit;
for(rit = vec.rbegin(); rit != vec.rend(); ++rit) {
     std::cout << *rit << " ";
}
```