# Estructuras

# pair

#include <utility>

- Almacena **2** valores.
- Declaración:

```
pair<tipo1, tipo2> nombre;
```

- Se puede acceder al primer valor con **`first`**.
- Se puede acceder al segundo valor con **`second`**.

# tuples

#include <tuple>

- Almacena **varios** tipos.
- Declaración:

  `tuple<tipo1, tipo2, ..., tipon> nombre;`

- Para acceder al elemento `i`:

  `get<tipoi>(nombre);`

# map

#include <map>

- Similar a los diccionarios en *python*.
- Almacena **llave** (**key**) y **valor** (**value**) y los **ordena** por la **llave**.
- Internamente usa un árbol binario.
- No permite llaves repetidas.
- Declaración:

  `map<tipo_llave, tipo_valor> nombre;`

- Para acceder al valor de una **`llave`**:

  `nombre[“llave”];`

# unordered_map

#include <unordered_map>

- Similar a los diccionarios en *python*.
- Almacena **llave** (***key***) y **valor** (***value***).
- Internamente usa *hashes*.
- No permite llaves repetidas.
- Declaración:

```
unordered_map<tipo_llave, tipo_valor> nombre;
```

- Para acceder al valor de una **`llave`**:

```
nombre[“llave”];
```

# set

#include <set>

- Almacena un valor.
- Está ordenado.
- Internamente usa un árbol binario.
- No permite valores repetidos.
- Declaración:

```
set<tipo> nombre;
```

# unrodered_set

#include <unordered_set>

- Almacena un valor.
- Internamente usa un árbol binario.
- No permite valores repetidos.
- Declaración:

```
unordered_set<tipo>
nombre;
```