Programación Orientada a Objetos

Práctica 5 – Marketplace. La clase Basket.

Mapas

- Son contenedores que almacenan asociaciones de clave (key) y valor (value) de manera ordenada según su clave.
- Los valores de clave identifican de manera única los elementos almacenados.
  - Esto es, una clave aparece una única vez en el mapa!
- Clave y valor pueden tener tipos de datos diferentes.

Mapas

- En C++ podemos usarlos a través de la plantilla std::map.
  - #include <map>
- Al crear un mapa, especificaremos los tipos de dato para clave y valor.
  - std::map<**tipo1,tipo2**> mapa_ejemplo_;
- Podemos añadir un elemento al mapa usando como índice su clave.
  - `mapa_ejemplo_[clave] = valor;`

Mapas

- Si repetimos clave en sucesivas asignaciones sobreescribimos el valor previamente asignado.
  - mapa_ejemplo_[clave] = valor actualiza el valor que acompaña a clave, si clave ya estaba presente en el mapa.
- Podemos eliminar elementos a partir de su clave:
  - mapa_ejemplo_.erase(clave);

Listas

- Es un contenedor que nos permite almacenar elementos en forma de secuencia.
- Cada elemento guarda referencias al anterior y al siguiente.
- Insertar y borrar elementos es menos costoso que usando vectores.
  - Se usa memoria no-contigua.
- Acceder a un elemento determinado es costoso.
  - El acceso es secuencial, no aleatorio.

Listas

- En C++ podemos usarlos a través de la plantilla std::list.
  - #include <list>
- Al crear una lista, especificaremos el tipo de dato de sus elementos.
  - std::list<**tipo**> lista_ejemplo_;
- Podemos añadir un elemento al final de manera sencilla:
  - lista_ejemplo_.push_back(elemento);

Listas

- Para la navegación a lo largo de la lista podemos ayudarnos de iteradores:
  - for (**auto** iterador=lista_ejemplo_.begin(); iterador!=lista_ejemplo_.end(); iterador++)
    ...
    }
  - Los iteradores actúan como un “puntero” a un elemento de la lista.
    - Accedemos a sus campos/métodos usando ->: iterador->Método();
- También podemos usarlos para eliminar elementos de la lista:
  - lista_ejemplo_.erase(iterador);

Ejercicio

- Crear la clase Basket, que representa una cesta de productos:
  - Se almacenan los productos en la cesta mediante una lista.
  - Las cantidades de productos se contabilizan usando un mapa.
  - Se almacena el coste total de los productos.
- Añadir los métodos para añadir y eliminar productos de la cesta, así como la cesta completa.
- Desarrollar un pequeño programa principal para probar nuestra implementación.
- Finalmente, pasar los test disponibles en Moodle.