PISTAS

1. Necesitas definir dos contenedores diferentes de tipos diferentes. Una forma de hacerlo es utilizando un vector de enteros y una lista de cadenas, como en el ejemplo de código a continuación:

std::vector<int> numeros = { 5, 2, 7, 1, 8 }; std::list<std::string> palabras = { "perro", "gato", "elefante", "leon", "tigre" };

1. Ejemplo de cómo definir una lista vacía y agregar algunas cadenas a la misma:

std::list<std::string> palabras; palabras.push\_back("manzana"); palabras.push\_back("banana"); palabras.push\_back("pera");

1. Ejemplo de cómo ordenar un vector utilizando **std::sort** y **std::greater**:

std::vector<int> numeros = { 5, 2, 7, 1, 8 }; std::sort(numeros.begin(), numeros.end(), std::greater<int>());

1. Ejemplo de cómo ordenar un vector utilizando **std::make\_heap**, **std::sort\_heap** y **std::greater**:

std::vector<int> numeros = { 5, 2, 7, 1, 8 }; std::make\_heap(numeros.begin(), numeros.end(), std::greater<int>()); std::sort\_heap(numeros.begin(), numeros.end(), std::greater<int>());

1. Cómo utilizar un iterador para recorrer el contenedor ordenado y mostrar sus elementos. En este caso, el código dentro del ciclo for puede imprimir los elementos ordenados del contenedor utilizando el operador **\*** para desreferenciar el iterador y acceder a los elementos individuales. Finalmente, puedes imprimir los elementos ordenados utilizando un ciclo for y un iterador para cada contenedor.

for (auto iter = numeros.begin(); iter != numeros.end(); ++iter) { std::cout << \*iter << " "; } std::cout << std::endl; for (auto iter = palabras.begin(); iter != palabras.end(); ++iter) { std::cout << \*iter << " "; } std::cout << std::endl;