- Solution

In this lesson, we will go over the solution to initializing containers using an initializer list.

WE'LL COVER THE FOLLOWING

Solution

```
// initializerList.cpp
#include <array>
#include <iostream>
#include <set>
#include <unordered_set>
#include <vector>

int main(){

  std::cout << std::endl;
  std::array<int, 5> myArray = {-10, 5, 1, 4, 5};
  for (auto i: myArray) std::cout << i << " ";
  std::cout << "\n\n";

  std::vector<int> myVector = {-10, 5, 1, 4, 5};
  for (auto i: myVector) std::cout << i << " ";
  std::cout << "\n\n";

  std::set<int> mySet = {-10, 5, 1, 4, 5};
  for (auto i: mySet) std::cout << i << " ";
  std::cout << "\n\n";

  std::unordered_multiset<int> myUnorderedMultiSet = {-10, 5, 1, 4, 5};
  for (auto i: myUnorderedMultiSet) std::cout << i << " ";
  std::cout << "\n";

  std::cout << std::endl;

}
```

Explanation

- In line 11, an `std::array`, of size 5 and type integers, is created with the given data.
- In line 15, an `std::vector` is created, of type integers, with the given data. Integers are not inserted in numerical order.
- In line 19, an `std::set` is created, of type integers, with the given data. Integers are inserted in numerical order and duplicate elements (in this case 5) are not inserted in the set.
- In line 23, an `std::unordered_multiset` is created, of type integers, with the given data. The keys are not sorted, and duplicate keys are allowed in `std::unordered_multiset`.

For further information, see intializer_list.

In the next lesson, we will learn about automatic type deductions using the `auto` keyword.