

Iterator Overview Solutions

- Explain what is meant by an iterator
 - An iterator is associated with a type (usually a container)
 - It represents the position of an element in the container
 - It can be used to “step through” the elements of the container
 - It can be dereferenced to access the value of the current element

- Describe the function of the `begin()` and `end()` member functions of a C++ standard container
 - `begin()` returns an iterator positioned at the first element
 - `end()` returns an invalid iterator. This represents the result of iterating past the last element in the container
 - If the container is empty, `begin()` and `end()` will return the same iterator

- Describe some operators that can be used with iterators
 - Dereference (* and ->) to access element values
 - Increment (++) to move forwards one element
 - Equality (==) and inequality (!=) to determine whether two iterators represent the same element

- Write a simple program that populates a vector and uses an iterator loop to print out all its elements

- What is an "iterator range"?
 - An iterator range is defined by a pair of iterators
 - The range is “half-open”, meaning it includes the first iterator of the pair, but not the second
 - e.g. `begin()` and `end()` define an iterator range from the first element up to, but not including, the last-plus-one element
 - In other words, from the first element to the last element