Container

Vector

Single element: Each element of single value

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/vector_element_1.cpp)
- methods
  - [x] insert
  - [x] access
  - [x] search
  - [x] display

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/vector_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Multiple element: Each element of multiple values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/vector_element_mul.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

List

Single element: Each element of single value

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/list_element_1.cpp)
- methods
  - [x] insert
  - [x] access
  - [x] search
  - [x] display

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/list_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Multiple element: Each element of multiple values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/list_element_mul.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Deque

Single element: Each element of single value

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/deque_element_1.cpp)
- methods
  - [x] insert

- [x] access
- [x] search
- [x] display

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/deque_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Multiple element: Each element of multiple values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/deque_element_mul.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Set

Single element: Each element of single value

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/set_element_1.cpp)
- methods
  - [x] insert
  - [x] access
  - [x] search
  - [x] display

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/set_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Multiple element: Each element of multiple values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/set_element_mul.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Multiset

Single element: Each element of single value

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/multiset_element_1.cpp)
- methods
  - [x] insert
  - [x] access
  - [x] search
  - [x] display

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/multiset_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search

- [x] display

Map

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/map_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Multimap

Two element: Each element of two values

- Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/map_element_2.cpp)
- methods
  - [x] insert
  - [x] access
  - [ ] search
  - [x] display

Bonus

- Convert list to vector
  - Why?
    - bcoz, difficult to access element by position, unlike vector, deque.
    - non-contiguous memory storage
  - Element type (single)
  - How?
    - simply add the list begineend() into vector function.
  - Example code (/F:/Coding/github_repos/cpp-playground/libs/boost/examples/container/bonus-1_convert_list_to_vector.cpp)

Facts: - If you frequently need to access the Nth element of a sequence, std::list, which is implemented as a doubly linked list, is probably not the right choice. std::vector or std::deque would likely be better. - `std::list` doesn't provide any function to get element given an index. You may try to get it by writing some code, which I wouldn't recommend, because that would be inefficient if you frequently need to do so. - you can get an iterator to the Nth element using `std::advance` or `std::next`

```
if (l.size() > N)
{
    std::list<Object>::iterator it = std::next(l.begin(), N);
}
```

- Maybe not the most efficient way. But you could convert the list into a vector.

```
std::vector<int> v1(l1.begin(), l1.end());
std::cout << v1[0] << std::endl;
```