C++ REVIEW

ITERATORS
STANDARD TEMPLATE LIBRARY

• A general purpose library of data structures (containers) and algorithms communicating through iterators.

• Plug each piece together and use in different applications.

```cpp
#include <vector>
#include <algorithm>

int main(){
    int myArray[4] = {2, 8, 5, 3};
    std::vector<int> v(myArray, myArray+4);
    std::sort(v.begin(), v.end());
}
```
CONTAINERS

• Container is an object storing other objects which can be accessed by iterators.

• Let $X$ be a container, then some important associated types:
  • $X::\text{value\_type}$ – type of data stored in the container
  • $X::\text{iterator}$ – type of iterator for the container
  • $X::\text{const\_iterator}$
  • ...
CONTAINERS

• Valid expressions
  • \( a.\text{begin}(\ ) \) – beginning of container
  • \( a.\text{end}(\ ) \) – one past the last element of container
  • \( a.\text{size}(\ ) \) – size of container
  • \( a.\text{empty}(\ ) \) – does the container have anything in it?
ITERATORS

• Iterators are used to point to other objects.

• Iterators play an interface between data structures and algorithms

• Main operations
  • advance(), next(), operator ++
  • Dereference – operator * 0, operator → 0
ITERATORS

• Forward Iterator – Only allows traversal from begin to end
• Bidirectional Iterator – Allows traversal in both directions (reversed order)
• Random Access Iterator – arbitrarily access elements in any order
• etc.

• Algorithms are defined for certain types of iterators, i.e., sort is only defined for random access iterating
EXERCISE

• Create a vector containing single character values
• Iterate over the vector and output each character
• Reverse iterate over the vector and output each character
• Sort the vector (uses random access iteration)
• Iterate over the vector using integers to output each character using random access iterators
  • i.e., using for(size_t i = 0; i < v.size(); ++i) and *(v.begin() + i)