**Problem 1**

**IntVector.cpp**

#include "IntVector.h"

#include <iostream>

using namespace std;

IntVector::IntVector(const int aArrayOfIntegers[], size\_t aNumberOfElements)

{

fNumberOfElements = aNumberOfElements;

fElements = new int[fNumberOfElements];

for (size\_t i = 0; i < fNumberOfElements; i++) {

fElements[i] = aArrayOfIntegers[i];

}

}

IntVector::~IntVector()

{

delete fElements;

}

size\_t IntVector::size() const {

return fNumberOfElements;

}

void IntVector::swap(size\_t aSourceIndex, size\_t aTargetIndex) {

if (aSourceIndex == aTargetIndex) {

throw out\_of\_range("Can't swap the same index.");

}

if (aSourceIndex >= fNumberOfElements) {

throw out\_of\_range("Illegal Source Vector Index.");

}

if (aTargetIndex >= fNumberOfElements) {

throw out\_of\_range("Illegal Target Vector Index.");

}

int lBuffer = fElements[aSourceIndex];

fElements[aSourceIndex] = fElements[aTargetIndex];

fElements[aTargetIndex] = lBuffer;

}

void IntVector::sort(const IntSorter& aSorter) {

aSorter(\*this);

}

const int IntVector::operator[](size\_t aIndex) const {

if (aIndex < 0 || aIndex >= fNumberOfElements) {

throw out\_of\_range("Illegal Vector Index.");

}

return fElements[aIndex];

}

IntVectorIterator IntVector::begin() const {

return IntVectorIterator (\*this);

}

IntVectorIterator IntVector::end() const {

return IntVectorIterator(\*this, size());

}

**IntVectorIterator.cpp**

#include "IntVector.h"

#include "IntVectorIterator.h"

IntVectorIterator::IntVectorIterator(const IntVector& aContainer, size\_t aStart) : fContainer(aContainer), fPosition(aStart) {

}

const int IntVectorIterator::operator\*() const {

return fContainer[fPosition];

}

IntVectorIterator& IntVectorIterator::operator++() {

fPosition++;

return \*this;

}

IntVectorIterator IntVectorIterator::operator++(int) {

IntVectorIterator old = \*this;

++(\*this);

return old;

}

bool IntVectorIterator::operator==(const IntVectorIterator& aRHS) const {

return

&fContainer == &aRHS.fContainer &&

fPosition == aRHS.fPosition;

}

bool IntVectorIterator::operator!=(const IntVectorIterator& aRHS) const {

return !(\*this == aRHS);

}

IntVectorIterator IntVectorIterator::begin() const {

IntVectorIterator iter = \*this;

iter.fPosition = 0;

return iter;

}

IntVectorIterator IntVectorIterator::end() const {

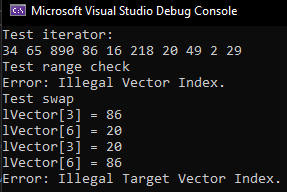
IntVectorIterator iter = \*this;

iter.fPosition = iter.fContainer.size();

return iter;

}

**Output**

****

**CocktailShakerSort.cpp**

#include "IntVector.h"

#include "CocktailShakerSort.h"

void CocktailShakerSort::operator()(IntVector& aContainer) const {

int beginIndex = 0;

int endIndex = aContainer.size() - 1;

while (beginIndex < endIndex) {

for (int i = beginIndex; i <= endIndex - 1; i++)

{

if (aContainer[i] > aContainer[i + 1]) {

aContainer.swap(i, i + 1);

}

}

endIndex--;

for (int i = endIndex; i >= beginIndex + 1; i--)

{

if (aContainer[i] < aContainer[i - 1]) {

aContainer.swap(i, i - 1);

}

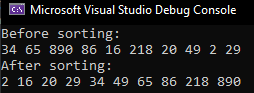
}

beginIndex++;

}

}

**Output**

****