```cpp
#include "IntVector.h"
#include <cstddef>
#include <stdexcept>
// #include <type_traits>
#include <utility>
// #include <vcruntime_new.h>

IntVector::IntVector(const int aArrayOfIntegers[], size_t aNumberOfElements) {
    this->fNumberOfElements = aNumberOfElements;
    this->fElements = new int[aNumberOfElements];
    for (int i = 0; i < aNumberOfElements; i++) {
        this->fElements[i] = aArrayOfIntegers[i];
    }
}

IntVector::~IntVector() {
    // delete this->fNumberOfElements;
    delete[] this->fElements;
}

size_t IntVector::size() const { return this->fNumberOfElements; }

const int IntVector::operator[](size_t aIndex) const {
    // we don't need the *this here, or am I missing something?
    if (aIndex < 0 || aIndex >= this->fNumberOfElements) {
        throw std::out_of_range("Illegal vector index");
    }
    return this->fElements[aIndex];
}

const int IntVector::get(size_t aIndex) const {
    // I guess this is where it actually goes?
    return (*this)[aIndex];
}

void IntVector::swap(size_t aSourceIndex, size_t aTargetIndex) {
    if (aSourceIndex < 0 || aSourceIndex >= this->fNumberOfElements ||
        aTargetIndex < 0 || aTargetIndex >= this->fNumberOfElements) {
        throw std::out_of_range("Illegal vector indicies");
    }

    // yes this function actually exists
    std::swap(this->fElements[aSourceIndex], this->fElements[aTargetIndex]);
}
```