```cpp
// Problem Set 2, 2024

#include <stdexcept>
#include "IntVector.h"

IntVector::IntVector(const int aArrayOfIntegers[], size_t aNumberOfElements) :
    // member initializer
    fNumberOfElements(aNumberOfElements)
{
    // creates a dynamic array of int
    fElements = new int[fNumberOfElements];

    for (size_t i = 0; i < aNumberOfElements; i++)
    {
        fElements[i] = aArrayOfIntegers[i];
    }
}

IntVector::~IntVector()
{
    // releases memory of the dynamic array
    delete[] fElements;
}

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

const int IntVector::get(size_t aIndex) const
{
    // reuse operator[] to checks index
    return (*this)[aIndex];
}

void IntVector::swap(size_t aSourceIndex, size_t aTargetIndex)
{
    // checks if indices are in range
    // we could also reuse operator[] to check each element
    if ((aSourceIndex >= fNumberOfElements) ||
        (aTargetIndex >= fNumberOfElements))
    {
        throw std::out_of_range("Illegal vector indices!!");
    }

    int lSourceElement = fElements[aSourceIndex];
    fElements[aSourceIndex] = fElements[aTargetIndex];
    fElements[aTargetIndex] = lSourceElement;
}

const int IntVector::operator[](size_t aIndex) const
{
    // checks if index is in range
    if (aIndex >= fNumberOfElements)
    {
        throw std::out_of_range("Illegal vector index!");
    }

    return fElements[aIndex];
}
```