Swinburne University of Technology

Faculty of Science, Engineering and Technology

ASSIGNMENT COVER SHEET

Subject Code: COS30008
Subject Title: Data Structures and Patterns
Assignment number and title: 2, Indexers, Method Overriding, and Lambdas
Due date: April 7, 2022, 14:30
Lecturer: Dr. Markus Lumpe

Your name: Nguyen Minh Duy **Your student id:** 104974743

Check Tutorial	Mon 10:30	Mon 14:30	Tues 08:30	Tues 10:30	Tues 12:30	Tues 14:30	Tues 16:30	Wed 08:30	Wed 10:30	Wed 12:30	Wed 14:30

Marker's comments:

Problem	Marks	Obtained
1	48	
2	30+10= 40	
3	58	
Total	146	

Extension certification:

This assignment has been given an extension and is now due on _____

Signature of Convener:_____

Problem 1 – IntVector.cpp


```cpp
#include "IntVector.h"
#include <stdexcept>  // Handle exceptions

using namespace std;

// Constructor: Initialize with an array of integers and its size
IntVector::IntVector(const int aArrayOfIntegers[], size_t aNumberOfElements)
{
    fNumberOfElements = aNumberOfElements;  // Set size
    fElements = new int[fNumberOfElements];  // Allocate memory

    // Copy elements to the internal array
    for (size_t i = 0; i < fNumberOfElements; i++)
    {
        fElements[i] = aArrayOfIntegers[i];
    }
}

// Destructor: Release allocated memory
IntVector::~IntVector()
{
    delete[] fElements;  // Free memory
}

// Return the size of the vector
size_t IntVector::size() const
{
    return fNumberOfElements;  // Return size
}

// Get the element at a specific index
const int IntVector::get(size_t aIndex) const
{
    return (*this)[aIndex];  // Return element at index
}

// Swap two elements by their indices
void IntVector::swap(size_t aSourceIndex, size_t aTargetIndex)
{
    // Ensure indices are within bounds
    if (aSourceIndex < fNumberOfElements && aTargetIndex < fNumberOfElements)
    {
        int lTemp = fElements[aSourceIndex];  // Temp store
        fElements[aSourceIndex] = fElements[aTargetIndex];  // Swap
        fElements[aTargetIndex] = lTemp;  // Complete swap
    }
    else
```

```cpp
    {
        throw out_of_range("Illegal vector indices");  // Handle out-of-bounds access
    }
}

// Overload [] to access elements at a given index
const int IntVector::operator[](size_t aIndex) const
{
    if (aIndex < fNumberOfElements)
    {
        return fElements[aIndex];  // Return element
    }

    throw out_of_range("Illegal vector index");  // Handle invalid index
}

```

Output:

```
Microsoft Visual Studio Debu

Test range check:
Properly caught error: Illegal vector index
Test swap:
lVector[3] = 86
lVector[6] = 20
lVector.get( 3 ) = 20
lVector.get( 6 ) = 86
Properly caught error: Illegal vector indices

D:\0Study\0C30008 Data Structures_And_Patterns\Problem Set 2\Project2\x64\Debug\Project2.exe (process 48360) exited with
 code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Problem 2 – SortableIntVector.cpp


```cpp
#include "SortableIntVector.h"

// Constructor: Initialize using base class IntVector
SortableIntVector::SortableIntVector(const int aArrayOfIntegers[], size_t aNumberOfElements) :
    IntVector(aArrayOfIntegers, aNumberOfElements)
{}

// Sort function using a comparison function to define the order
void SortableIntVector::sort(Comparable aOrderFunction)
{
    // Outer loop through all elements
    for (size_t i = 0; i < size(); i++)
    {
        // Inner loop to compare and swap elements
        for (size_t j = size() - 1; j > i; j--)
        {
            if (aOrderFunction(get(j), get(j - 1)))  // Compare elements
            {
                swap(j, j - 1);  // Swap elements if needed
            }
        }
    }
}

```

Output:

```
Microsoft Visual Studio Debu

Bubble Sort:
Before sorting:
34 65 890 86 16 218 20 49 2 29
After sorting:
2 16 20 29 34 49 65 86 218 890

D:\0Study\0C30008 Data Structures_And_Patterns\Problem Set 2\Project2\x64\Debug\Project2.exe (process 11280) exited with
 code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Problem 3 – ShakerSortableIntVector.cpp


```cpp
#include "ShakerSortableIntVector.h"

// Constructor, initializes using base class constructor
ShakerSortableIntVector::ShakerSortableIntVector(const int aArrayOfIntegers [], size_t aNumberOfElements) :
    SortableIntVector(aArrayOfIntegers, aNumberOfElements)
{}

// Shaker sort function, sorts elements using a comparison function
void ShakerSortableIntVector::sort(Comparable aOrderFunction)
{
    size_t lBeginIndex = 0;  // Start of the unsorted section
    size_t lEndIndex = size() - 1;  // End of the unsorted section

    // Continue sorting while there are unsorted elements
    while (lBeginIndex < lEndIndex)
    {
        // Forward pass to push the largest element to the end
        for (size_t i = lBeginIndex; i < lEndIndex; i++)
        {
            if (aOrderFunction(get(i), get(i + 1)))
            {
                swap(i, i + 1);  // Swap adjacent elements
            }
        }

        lEndIndex--;  // Reduce the unsorted section from the end

        // Backward pass to push the smallest element to the beginning
        for (size_t i = lEndIndex; i > lBeginIndex; i--)
        {
            if (aOrderFunction(get(i - 1), get(i)))
            {
                swap(i - 1, i);  // Swap adjacent elements
            }
        }

        lBeginIndex++;  // Reduce the unsorted section from the beginning
    }
}

```

Output:

```
Microsoft Visual Studio Debu

Cocktail Shaker Sort:
Before sorting:
34 65 890 86 16 218 20 49 2 29
After sorting:
890 218 86 65 49 34 29 20 16 2

D:\0Study\0C30008 Data Structures_And_Patterns\Problem Set 2\Project2\x64\Debug\Project2.exe (process 27392) exited with
 code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Main_PS2 - line 93 added


```
1  #include <iostream>
2  #include <stdexcept>
3
4  using namespace std;
5
6  //#define P1
7  //#define P2
8  #define P3
9
10 #ifdef P1
11
12 #include "IntVector.h"
13
14 void runP1()
15 {
16     int lArray[] = { 34, 65, 890, 86, 16, 218, 20, 49, 2, 29 };
17     size_t lArrayLength = sizeof(lArray) / sizeof(int);
18
19     IntVector lVector( lArray, lArrayLength );
20
21     cout << "Test range check:" << endl;
22
23     try
24     {
25         int lValue = lVector[lArrayLength];
26
27         cerr << "Error, you should not see " << lValue << " here!" <<
             endl;
28     }
29     catch (out_of_range e)
30     {
31         cerr << "Properly caught error: " << e.what() << endl;
32     }
33     catch (...)
34     {
35         cerr << "This message must not be printed!" << endl;
36     }
37
38     cout << "Test swap:" << endl;
39
40     try
41     {
42         cout << "lVector[3] = " << lVector[3] << endl;
43         cout << "lVector[6] = " << lVector[6] << endl;
44
45         lVector.swap( 3, 6 );
46
47         cout << "lVector.get( 3 ) = " << lVector.get( 3 ) << endl;
48         cout << "lVector.get( 6 ) = " << lVector.get( 6 ) << endl;
```

```cpp

        lVector.swap( 5, 20 );

        cerr << "Error, you should not see this message!" << endl;
    }
    catch (out_of_range e)
    {
        cerr << "Properly caught error: " << e.what() << endl;
    }
    catch (...)
    {
        cerr << "Error, this message must not be printed!" << endl;
    }
}

#endif

#ifdef P2

#include "SortableIntVector.h"

void runP2()
{
    int lArray[] = { 34, 65, 890, 86, 16, 218, 20, 49, 2, 29 };
    size_t lArrayLength = sizeof(lArray) / sizeof(int);

    SortableIntVector lVector( lArray, lArrayLength );

    cout << "Bubble Sort:" << endl;

    cout << "Before sorting:" << endl;

    for ( size_t i = 0; i < lVector.size(); i++ )
    {
        cout << lVector[i] << ' ';
    }

    cout << endl;

    // Use a lambda expression here that orders integers in increasing order.
    // The lambda expression does not capture any variables of throws any exceptions.
    // It has to return a bool value.
//    lVector.sort( /* lambda expression */ );

    lVector.sort( [](int aLeft, int aRight ) { return aLeft <= aRight; } );         // new line

```

```
    cout << "After sorting:" << endl;

    for ( size_t i = 0; i < lVector.size(); i++ )
    {
        cout << lVector[i] << ' ';
    }

    cout << endl;
}

#endif

#ifdef P3

#include "ShakerSortableIntVector.h"

void runP3()
{
    int lArray[] = { 34, 65, 890, 86, 16, 218, 20, 49, 2, 29 };
    size_t lArrayLength = sizeof(lArray) / sizeof(int);

    ShakerSortableIntVector lVector( lArray, lArrayLength );

    cout << "Cocktail Shaker Sort:" << endl;

    cout << "Before sorting:" << endl;

    for ( size_t i = 0; i < lVector.size(); i++ )
    {
        cout << lVector[i] << ' ';
    }

    cout << endl;

    // sort in decreasing order
    lVector.sort();

    cout << "After sorting:" << endl;

    for ( size_t i = 0; i < lVector.size(); i++ )
    {
        cout << lVector[i] << ' ';
    }

    cout << endl;
}

#endif

```

```
int main()
{
#ifdef P1

    runP1();

#endif

#ifdef P2

    runP2();

#endif

#ifdef P3

    runP3();

#endif

    return 0;
}

```