**Assessed Practical 2**

**15th November 2017**

In Microsoft Visual Studio create a new Empty Project called “Assessed\_Practical2” on the C:\ drive or on your own drive. Add a new source file “assessed\_main.cpp” to the project.

There are two files: uConvert.h and fragments.txt in Queens Online. Look in ResourcesPractical BookletsCSC2040 Assessed Practical 2 or in the Assignment Tool under CSC2040 Assessed Practical 2. “fragments.txt” contains some pieces of code mentioned in Parts 3 and 4 below, to save you typing these all over again. Open “fragments.txt” using a text editor. Copy and paste whichever bits you need.

**Part 1 Copy Constructor (5 marks)**

Inside Project Assessed\_Practical2, follow Steps 1-3 below to create a C++ program:

1. Create a header file “xarray.h”, in which you declare a class xArray:

class xArray {

public:

xArray(int size);

~xArray();

int size;

int\* data;

};

2. Create a source file “xarray.cpp”, in which you type the code below to implement this class:

#include"xArray.h"

xArray::xArray(int size)

: size(size)

{

data = new int[size];

}

xArray::~xArray()

{

delete[] data;

}

3. In the source file “assessed\_main.cpp”, you type the program below for testing the class:

#include <iostream>

#include "xarray.h"

using namespace std;

int main()

{

xArray array(3);

for (int i = 0; i < 3; i++) array.data[i] = i;

if (array.size > 0) {

xArray array2 = array;

}

cout << array.data[0];

return 0;

}

**Problem:**

Run the above program and you will see that it prints out a random number instead of the expected 0, and most likely crashes. You are asked to add a copy constructor for the class to fix this problem.

**Part 2 Operator Overloading (5 marks)**

Continue with the above xArray class. By our definition, one object a of class xArray is greater than or equal to (>=) another object b of class xArray if the sum of the values in the data array of object a is greater than or equal to the sum of the values in the data array of object b. For example, the following two objects a and b satisfy a >= b:

xArray a(2);

a.data[0] = 10; a.data[1] = 10;

xArray b(3);

b.data[0] = 4; b.data[1] = 9; b.data[2] = 5;

because the sum of the values of a’s data array (20) is greater than the sum of the values of b’s data array (18).

**Problem**

In the above class xArray add an operator function to overload the >= operator for comparing two objects of the class based on the above definition. Then, in the same source file “assessed\_main.cpp” rename the main function for Part 1 to main1 and create a new main function to test that your >= operator is working as expected for comparing two objects (as the above). You can achieve this by writing two if statements: one comparing a >= b and print out “a >= b is true”, and the other comparing b >= a and print out “b >= a is false”.

**Part 3 Template Function (5 marks)**

The following shows a C++ function that can be used to remove a value at a position (pos) from an integer array.

void remove(int\* &array, int &array\_len, int pos)

{

if (!array || array\_len <= 1) {

cout << "invalid array" << endl;

return;

}

if (pos < 0 || pos >= array\_len) {

cout << "pos is out of range" << endl;

return;

}

// new array length after removal

array\_len--;

// new array

int\* temp = new int[array\_len];

for (int i = 0; i < pos; i++) temp[i] = array[i];

for (int i = pos; i < array\_len; i++) temp[i] = array[i + 1];

// delete original array

delete[] array;

// set the new array

array = temp;

}

**Problem:**

You are asked to convert the above function into a template function so that it can be applied to other types of arrays. Do this inside the same source file “assessed\_main.cpp”. First rename the main function for Part 2 to main2. Then type or copy the code for the template function. Finally below the template function create a new main function to do the following to test the template function you have created:

1. Declare an int variable array\_len and initialize it to a value of 10.
2. Allocate a double array of size array\_len by using new.
3. Fill the array with arbitrary values.
4. Print the values of the array.
5. Call the template function remove to remove the value at position 5.
6. Print the values of the new array.
7. Repeat Steps 1-6 to test the function with a char array.

**Part 4 Inheritance and Virtual Function (5 marks)**

Place the provided file uConvert.h into the folder Assessed\_Practical2\ Assessed\_Practical2, and #include it in the “assessed\_main.cpp”.

uConvert.h is an attempt to combine class inheritance and virtual functions to implement a form of “one interface, multiple methods” (i.e., polymorphism) for the conversion of values from one unit to another. The following main function shows an example of how to use this converter.

int main()

{

// a base class pointer

uConvert\* base\_ptr = 0;

// menu to take input from the users

cout << "\nEnter 1 for litre to gallon conversion\n"

<< "Enter 2 for Fahrenheit to Celsius conversion\n"

<< "Enter 3 for feet to meter conversion\n";

int option;

cin >> option;

// pointing base\_ptr to the derived object of the user's choice

switch (option) {

case 1:

base\_ptr = new l2g;

break;

case 2:

base\_ptr = new f2c;

break;

case 3:

base\_ptr = new f2m;

break;

default:

return 0;

}

// invoking the conversion

base\_ptr->read\_unit();

base\_ptr->convert();

base\_ptr->print();

delete base\_ptr;

return 0;

}

**Problem:**

(1) First in the source file “assessed\_main.cpp” rename the main function for Part 3 to main3. Then type (or copy) the above main function into the file and then run the program. You will see that the program fails to build because uConvert.h contains an error. Identity and fix the error.

(2) Extend this converter by adding a new derived class p2k for the conversion from pounds to kilograms (lbs to kg). Note that 1 lb = 0.453592 kg.

(3) Add the new pounds-to-kilograms conversion option in the main function so that it can be selected from the menu and executed as the other conversion options.

Submission Notes

1. Save all your files:

xArray.h and xArray.cpp

uConvert.h

assessed\_main.cpp.

1. Close Visual Studio and locate the folder Assessed\_Practical2 on the C:\ drive. ZIP this folder (by right clicking the folder and choosing ‘Send To’ and selecting Compressed (Zipped) folder).
2. Check that your .zip file contains your source files (double click on the .zip file. You do not need to extract the files again).
3. In case something goes wrong with your submission, first make a backup of your zip file on to a USB drive, and/or email it to yourself.
4. Upload the .zip file to Queens Online using the Assignment tool into CSC2040 Assessed Practical 2.