Advanced Programming C++ - Practicals

Work your way through the implementation of the following exercises. The suggested plan is as follows:

Practical 1 (Monday)	A, A
Practical 2 (Tuesday)	A, A, A, B
Practical 3 (Thursday)	A, A, B, B
Practical 4 (Friday)	A, B, B, C
Practical 5 (Monday)	A, B, C, C
Practical 6 (Tuesday)	B, C, C
Practical 7 (Thursday)	A, B, C, C, C

Question A.1

Consider the following strings: Tokyo, Berlin, Rio, Denver, Helsinki, Nairobi. Create an STL container to store these strings. In a main, create the vector, sort the strings in the container into ascending alphabetic order, and display the strings to screen. Choose an appropriate container to perform these operations.

Question A.2

Consider the following integers: 9, 8, 7, 6, 5, 4, 3, 2, 1, 0. Use a container defined in the Standard Template Library to store these numbers. Write a single function that: removes all even numbers, sorts the numbers in the container into ascending order, and displays the numbers to screen. Choose an appropriate container to perform these operations.

Question A.3

Write a **function** that takes in an integer and then reverses the order of the digits in the integer. For instance, changes “12345678” to “87654321”. The main section of the program should allow the user to enter an integer, it should then call the function to reverse the integer’s order and display the resultant integer.

Question A.4

Write declarations for the following: a pointer to a character, an array of 10 integers, a reference to an array of 10 integers, a pointer to an array of character strings, a pointer to a pointer to a character, a constant integer, a pointer to a constant integer, and a constant pointer to an integer. Initialize each one.

Question A.5

You are a child minder and the children you are looking after would like to watch a movie. However, parents have instructed that the children should watch a movie that contains the word “happy” in the title and lasts less than 70 minutes. Write a programme that asks the user to enter a movie title and movie duration. The program should continue asking for a new movie title until you find one that meets the parent’s specification (i.e., contains “happy” and is <70 minutes long). Once you have found the appropriate movie, print an “accepted” and “rejected” list, which shows all of the previously entered movie titles.

Question A.6

Write a **function** that takes in a string and then reverses the order of the characters in the string. For instance, changes “diagonal” to “lanogaid”. The main section of the program should allow the user to enter a string, it should then call the function to reverse the string’s order and display the resultant string.

Question A.7

Write a function that swaps (exchanges the values of) two integers. Use **int*** as the argument type. Write another swap function using **int&** as the argument type.

Question A.8

Create a base class called Furniture with derived classes Chair, Table, and Sofa. Implement a virtual function price() in each derived class that calculates the price of the furniture based on its material and size. Store the material and sizes, and use them in your calculations.

Question A.9

Read (from a file or from the screen) a sequence of possibly whitespace-separated (name,value) pairs, for example:

```
John 20
Jane 21
Pippa 58
John 50
Pippa 10
```

where the name is a single whitespace-separated word (John) and the value is an integer or a floating-point value (20). Compute and print the sum and mean for each name (for Pippa = 58+10/2) and the sum and mean for all names ((20+21+58+50+10)/5).

Question A.10

Write a function that takes in a string and changes that string to Title Case. Title Case means that each letter should be lower case unless it is either the first letter in the string or if the preceding letter is a space, in which case the letter should be a capital. Allow the user to enter a string, use the function to convert the string to Title Case and then display the result. For example (where bold is user input):

```
Enter a string to convert: mAke ThIs tITle cAse

Output: Make This Title Case
```

Question A.11

Write a function that concatenates two strings. Use **a pointer** as the argument type. Write another swap function using **a reference** as the argument type.

Question A.12

Create a function that determines whether an integer is a prime number. The code to compute whether the number is prime does not have to be optimised. Use the following declaration:

```
bool isPrime(unsigned int n)
```

Write a programme that computes the values of the first thousand prime numbers. In a main, store these in an appropriate STL container so that they can be accessed rapidly. Display the first thousand prime numbers to screen, separated by comma values. Do not compute the prime numbers during the printing process, but rather print out the contents of your container.

Question B.1

Consider the following definition of ‘metallic means’.

The metallic means (also ratios or constants) of the successive natural numbers are the continued fractions:

$$n+\cfrac{1}{n+\cfrac{1}{n+\cfrac{1}{n+\cfrac{1}{n+\ddots}}}}=[n;n,n,n,n,\dots]=\frac{n+\sqrt{n^2+4}}{2}.$$

Metallic means (Metallic ratios)			Class
N	Ratio	Value	(Type)
0:	$\frac{0+\sqrt{4}}{2}$	1	
1:	$\frac{1+\sqrt{5}}{2}$	1.618033989[a]	Golden
2:	$\frac{2+\sqrt{8}}{2}$	2.414213562[b]	Silver
3:	$\frac{3+\sqrt{13}}{2}$	3.302775638[c]	Bronze
4:	$\frac{4+\sqrt{20}}{2}$	4.236067978[d]	
5:	$\frac{5+\sqrt{29}}{2}$	5.192582404[e]	
6:	$\frac{6+\sqrt{40}}{2}$	6.162277660[f]	
7:	$\frac{7+\sqrt{53}}{2}$	7.140054945[g]	
8:	$\frac{8+\sqrt{68}}{2}$	8.123105626[h]	
9:	$\frac{9+\sqrt{85}}{2}$	9.109772229[i]	
⋮			
n:	$\frac{n+\sqrt{4+n^2}}{2}$		

Create a function with the following declaration:

```
double computeMetallicMean(double n)
```

Write a programme that computes the values of the first ten metallic means as shown in the table.

Question B.2

Two quantities are in the **golden ratio** if their ratio is the same as the ratio of their sum to the larger of the two quantities. It follows that a and b are in the golden ratio if:

$$\frac{a+b}{a} = \frac{a}{b} = 1.618$$

Write a function that reads a single number, z, from the screen and computes two numbers, x and y, such that their sum is the original input number, $x + y = z$, and they are in the golden ratio, $x/y = 1.618$.

Question B.3

Write a programme that reads two vector of doubles from the screen. Store these two vectors using STL containers, and do not use pointers. Write a function that adds the two vectors without making a copy of the vectors. From a main function, print the entire operation to screen as follows:

```
QUESTION B.1:

VECTOR 1 = [1,2,3, 4.75,5,6, 1,-2,3]

VECTOR 2 = [1,0,0, 0,1,0, 0,0,1]

VECTOR 1 + VECTOR 2 = [2,2,3, 4.75,6,6, 1,-2,4]
```

Your programme should be able to add any vectors input by the user.

Question B.4

Write a programme that reads two 3x3 matrices from the terminal, in other words, you should write code to read matrices input from the standard input stream. Store these matrices using STL containers, and do not use pointers. Write a function that adds the two matrices without making a copy of the matrices. From your main function, print the entire operation to screen as follows:

```
MATRIX 1 = [1,2,3; 4.75,5,6; 1,-2,3]

MATRIX 2 = [1,0,0; 0,1,0; 0,0,1]

MATRIX 1 + MATRIX 2 = [2,2,3; 4.75,6,6; 1,-2,4]
```

Your programme should be able to add any two 3x3 matrices input by the user.

Question B.5

Read from a text file a sequence of (name, value, Boolean) truples, for example:

John	101.7	1
Jane	21	1
Pippa	58.78	0
John	50	0
Lawrence	103.98	1
Ellie-May	67.83	1

where the name is a single whitespace-separated word (John), the value is an integer or a floating-point value (101.7), and the Boolean is a 1 or a 0 value. Compute and print the sum and mean for all names for which the Boolean value is true (the mean would be = `(101.7+21+103.98+67.83)/4`), and the sum and mean for all names (the mean would be = `(101.7+21+58.78+50+103.98+67.83)/6`).

Question B.6

Write a programme that reads two strings from the screen. Write a function that concatenates the two strings and computes the length of the resulting concatenated string. This function should not make a copy of the strings. From a main function, print the entire operation to screen as follows:

```
QUESTION B.2:

STRING 1 = Johnny

STRING 2 = Rose

STRING 1 + STRING 2 = Johnny Rose

TOTAL LENGTH = 11
```

Your programme should be able to concatenate any two strings input by the user.

Question B.7

Read from a text file a sequence of (vegetable name, sow month, raised bed) tuples:

Leek	March	1
Carrots	April	1
Celery	April	1
Melon	March	1
Kohlrabi	February	2
Beans	March	2
Courgette	March	2
Nasturtiums	February	2
Cucumbers	April	3
Dill	April	3
Tomatoes	February	3
Nasturtiums	February	3

where the three entries on each line are separated by a space. Compute and print a list of vegetables that must be sown for each month, and report in which bed they must be sown. Your output to screen should be as follows:

```
Month: February
Vegetables to sow: Kohlrabi (Bed 2), Nasturtiums (Bed 2), Nasturtiums (Bed 3),
Tomatoes (Bed 3)
```

```
Month: March
Vegetables to sow: Leek (Bed 1), Melon (Bed 1), Beans (Bed 2), Courgette (Bed
2)

Month: April
Vegetables to sow: Carrots (Bed 1), Celery (Bed 1), Cucumbers (Bed 3), Dill
(Bed 3)
```

Question B.8

Generate 10,000 uniformly distributed random integers in the range 1 to 50,000 and store them in

1. a standard library `vector<int>`
2. a standard library `list<int>`
3. a standard library `set<int>`

Notice: At the end of the insertion each container should have the same 10,000 elements.

In each case, calculate the arithmetic mean of the elements of the vector (as if you didn't know it already). Time the resulting loops. Measure and report onto the screen the performance and memory consumption of the three containers.

Question B.9

Fibonacci numbers F_n, form a sequence, called the Fibonacci sequence, such that each number is the sum of the two preceding ones, starting from 0 and 1. That is,

$F_0 = 0$

$F_1 = 1$

and for $n > 1$:

$F_n = F_{n-1} + F_{n-2}$

Use standard library containers and functions to compute a sequence of the first 100 Fibonacci numbers. Do not use recursion (a recursive function is a function that calls itself). Instead, store Fibonacci numbers in a `std::map` as you compute them and reuse information in this map to compute future numbers. Print the 100 Fibonacci sequence to a text file.

Question B.10

A shop owner has found that her staff are not very good at working out the best way to give out change. Write a program that allows the staff to enter the amount an item cost, as well as the amount of money tendered. The program should then calculate the change to be given (number of £20, £10, £5 notes and £1, 50p, 20p, 10p, 5p, 2p and 1p coins).

The amounts should be entered as floating point `pounds.pence` amounts (e.g. 32.65 pounds). The computer should then convert these amounts into an integer number of pence (remember type casting).

Once the change in pence has been calculated, the program should work out the type of change to be given by giving as many of the larger notes or coins as possible. For example, `664p` change should be `1×£5, 1×£1, 1×50p, 0x20p, 1×10p, 0x5p, 2×2p` and `0x1p`. (A clue: Make use of integer maths)

Question B.11

Create a text file “ComputerProgramming.txt” with the following text:

```
Computer programming is the process of designing and building an executable computer
program for accomplishing a specific computing task. Programming involves tasks such as
analysis, generating algorithms, profiling algorithms' accuracy and resource consumption,
and the implementation of algorithms in a chosen programming language commonly referred to
as coding. The source code of a program is written in one or more programming languages.
The purpose of programming is to find a sequence of instructions that will automate the
performance of a task for solving a given problem. The process of programming thus often
requires expertise in several different subjects, including knowledge of the application
domain, specialized algorithms, and formal logic. Related programming tasks include
testing, debugging, maintaining a program's source code, implementation of build systems,
and management of derived artefacts such as machine code of computer programs. These might
be considered part of the programming process, but often the term software development is
used for this larger process with the term programming, implementation, or coding reserved
for the actual writing of source code. Software engineering combines engineering techniques
with software development practices.
```

Write a programme that reads a file, and produces a `std::map<string,int>` holding each string and the number of times that the string appears. Run the program on the “ComputerProgramming.txt” file and print out how many times each word appears in descending order.

Question B.12

Write a class that can store a 3x3 matrix of floating-point numbers. Write member functions that can (1) fill the array using user input, (2) display the matrix and (3) return the trace of the matrix (the trace of a square matrix is the sum of the elements in its diagonal).

The program should then randomly generate 100 different matrices and display the average of the traces of these matrices. Use dynamically allocated objects to store the matrices. Remember to free the memory when you are finished with it.

Question B.13

Consider a tetrahedron defined by four nodes a, b, c, d.

The volume of a tetrahedron with vertices $\mathbf{a} = (a_x, a_y, a_z)$, $\mathbf{b} = (b_x, b_y, b_z)$, $\mathbf{c} = (c_x, c_y, c_z)$, and $\mathbf{d} = (d_x, d_y, d_z)$, can be computed by solving this equation:

Equation C.1:

$$V = \frac{|(\mathbf{a}-\mathbf{d}) \cdot ((\mathbf{b}-\mathbf{d}) \times (\mathbf{c}-\mathbf{d}))|}{6}.$$

Now consider an irregular hexahedron defined by eight nodes h0, h1, h2, h3, h4, h5, h6, h7. The volume of an irregular hexahedron can be computed by adding the volume of sub-tetrahedra formed by vertices `(h0,h1,h3,h4), (h4,h1,h3,h5), (h4,h5,h3,h7), (h1,h2,h3,h6), (h3,h1,h6,h5), (h5,h6,h3,h7).`

Part 1: Write a function `VolumeOfTetra` that computes the volume of a tetrahedron using Equation C.1.

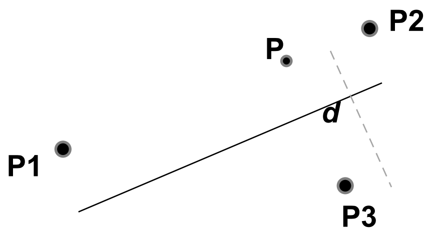
Part 2: Write a function `VolumeOfHexa` that computes the volume of a hexahedron by adding the volume of sub-tetrahedra that compose it.

Part 3: Write a main function that creates 1,000 random hexahedra and compute their volumes. Display the sum of the volumes of the computed hexahedra to screen. Note: ensure that your programme creates hexahedra that have eight distinct nodes.

Question B.14

The solution to finding the shortest distance from a point to a line segment is defined by the equation of a line defined through two points **P1** `(x1, y1)` and **P2** `(x2, y2)`

$$P = P1 + u(P2 - P1)$$



where

$$u = \frac{(x3 - x1)(x2 - x1) + (y3 - y1)(y2 - y1)}{||P2 - P1||^2}$$

And so, P = (x, y) can be defined as:

$$x = x1 + u(x2 - x1)y = y1 + u(y2 - y1)$$

The distance between the point **P3** and the line is the distance between **P** and **P3**. Write a function that computes the distance between a segment and a point. Write a programme (main function) that demonstrates the use of this distance function, and outputs the results to screen.

Note: ensure that your function will not crash if the two points are coincident.

For section C, download the code in GitHub labeled “assessment_matrix.zip”. This contains two classes: Matrix and CSRMatrix. You should modify these files to respond C Questions. You can include the methods in this question to the files used in Question E, but add comments identifying which code is for which question.

Question C.1

You are to implement a copy constructor in the CSRMatrix class. This method takes in a dense Matrix<T> as its only input and then initialises a sparse copy of the input matrix.

Create a CSRMatrix.cpp file and a cpp file containing a main method that populates a dense matrix that has some zero entries, then calls the copy constructor to build a new CSRMatrix and prints the resulting sparse matrix. When calculating the sparsity of the dense matrix, just ignore any entries with magnitude that are below 1x10^{-13}.

Question C.2

Write your own version of the std::shared_ptr<T> templated smart pointer, called *myShrdPtr* *(*note you cannot use the std::shared_ptr<T> in your implementation). This should be a templated class, where you are required to implement a constructor that takes a raw pointer as input and takes ownership of that memory.

You should also implement a copy constructor that takes an existing *myShrdPtr* and copies it. The *myShrdPtr* should keep track of how many other *myShrdPtr’s* point at the same object they own. You must include a destructor that deletes the memory owned by the *myShrdPtr* when there are no other *myShrdPtr* pointing at the same object. You must also overload the dereferencing operator, *, which returns the object myShrdPtr is pointing at by reference.

How would you change the implementation of *myShrdPtr* if this class were trying to implement std::weak_ptr? Write a few sentences describing your proposed changes (note you don’t have to make any changes or write extra code; be specific about your proposed changes, for example, would your dereferencing operator change?) in a document and modify the source files of *myShrdPtr.* You should also create a cpp file containing a main method that builds a new instance of *myShrdPtr.*

Question C.3

Part 1: You are to implement a method in the CSRMatrix class, called *copyDropValues*. This method takes a drop tolerance (a number of type T) and a dense Matrix<T> as input and returns a copy in a sparse format (i.e., it returns a CSRMatrix<T> object, either as a return type or as an argument) that only contains entries from the dense Matrix<T> that are bigger than the input drop tolerance.

You should try and minimise the amount of memory required to create this copy as best you can (e.g., **don’t** preallocate your sparse copy assuming that the nnzs = rows * cols).

For example, if a dense matrix of size 10x10 has the following entries:

```
15     0     2     0     4     0     6     0     8     0
10     0    12     0    14     0    16     0    18     0
20     0    22     0    24     0    26     0    28     0
30     0    32     0    34     0    36     0    38     0
40     0    42     0    44     0    46     0    48     0
50     0    52     0    54     0    56     0    58     0
60     0    62     0    64     0    66     0    68     0
70     0    72     0    74     0    76     0    78     0
80     0    82     0    84     0    86     0    88     0
90     0    92     0    94     0    96     0    98    30
```

and *copyDropValues* is called with a drop tolerance of 10, the resulting sparse matrix has *values*, *row_position* and *col_index* given by

values: [15 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 30]

row_position: [0 1 5 10 15 20 25 30 35 40 46]

col_index: [0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 0 2 4 6 8 9]

Part 2: Write a main method that instantiates a Matrix<T> object, populates it and prints it. The main method should then call the *copyDropValues* method on that object and print the resulting sparse CSRMatrix<T> matrix. The dense matrix you create and populate should have a number of values smaller than the drop tolerance used to test the *copyDropValues* method, i.e., a sparse matrix should be returned that has nnzs < row * cols.

Question C.4

You are to implement a method in the CSRMatrix class, called *dense2sparse*. This method creates a copy of the given dense Matrix<T> in a sparse format and hence returns a CSRmatrix<T> object (either as a return type or as an argument).

Modify your CSRMatrix.cpp file (and the .h if you have modified it) and create a cpp file containing a main method that instantiates a Matrix<T> object and populates it. This dense matrix should have a number of non zeros smaller than the number of rows times the number of columns, i.e., a matrix that has some zeros in it. The main method should then call the dense2sparse method on that object and print the resulting sparse CSRMatrix<T> matrix.

Question C.5

You are to implement a method in the CSRMatrix class, called *sparse2dense*. This method creates a copy of the given sparse CSRMatrix<T> in a dense format and hence returns a matrix<T> object.

Modify your your CSRMatrix.cpp file and create a cpp file containing a main method that instantiates a CSRMatrix<T> object and populates it. This sparse matrix should have a number of non zeros smaller than the number of rows times the number of columns, i.e., a matrix that has some zeros in it. The main method should then call the sparse2dense method on that object and print the resulting dense Matrix<T> matrix.

Question C.6

Part 1: You are to implement a method in the CSRMatrix<T> class, called *getInverseDiagonal*. This method returns a **shared** pointer of type T[], which points to the first entry of an array stored in memory on the heap, which contains the inverse of the diagonal of the matrix (i.e., the entries of the array are given by the inverse of each of the diagonal entries of the matrix). You should **not assume** the CSRMatrix is square (i.e., your method must work if the number of rows is not equal to the number of columns).

For example, if a matrix is of size 10x10 with the following entries:

```
    1.0000         0    0.4000         0    0.8000         0    1.2000         0    1.6000         0
    2.0000    2.0000    2.4000         0    2.8000         0    3.2000         0    3.6000         0
    4.0000         0    3.0000         0    4.8000         0    5.2000         0    5.6000         0
    6.0000         0    6.4000    4.0000    6.8000         0    7.2000         0    7.6000         0
    8.0000         0    8.4000         0    5.0000         0    9.2000         0    9.6000         0
   10.0000         0   10.4000         0   10.8000    6.0000   11.2000         0   11.6000         0
   12.0000         0   12.4000         0   12.8000         0    7.0000         0   13.6000         0
   14.0000         0   14.4000         0   14.8000         0   15.2000    8.0000   15.6000         0
   16.0000         0   16.4000         0   16.8000         0   17.2000         0    9.0000         0
   18.0000         0   18.4000         0   18.8000         0   19.2000         0   19.6000   10.0000
```

and *getInverseDiagonal* is called on the sparse version of this matrix, the entries returned within the array pointed at by the shared pointer are:

[1 0.5 0.333333 0.25 0.2 0.166667 0.142857 0.125 0.111111 0.1]

Part 2: Write a main method that instantiates a CSRMatrix<T> object, populates it and prints it. You must ensure the sparse matrix you create **does not** have any zeros on the diagonal. The sparse matrix you create should also have the number of non zeros smaller than rows x columns, i.e., a matrix that has some zeros in it. The main method should then call the *getInverseDiagonal* method and print the resulting array.

Question C.7

You are to implement a method in the CSRMatrix<T> class, called *getDiagonal*. This method returns a shared pointer of type T[], which points to the first entry of an array stored in memory on the heap, which contains the diagonal of the matrix.

Modify your CSRMatrix.cpp file (and the .h if you have modified it) and create a cpp file containing a main method that instantiates a CSRMatrix<T> object and populates it. This sparse matrix should have a number of non zeros smaller than the number of rows times the number of columns, i.e., a matrix that has some zeros in it. You should also not assume the CSRMatrix is square (i.e., your method must work if the number of rows is not equal to the number of columns). The main method should then call the getDiagonal method from the CSRMatrix and print the resulting array.

Question C.8

You are to implement a method in the CSRMatrix class, called *computeTrace*. This method computes the trace of the sparse matrix, which is the sum of all diagonal entries. This method should return only a double value containing the trace of the matrix.

Modify your CSRMatrix.cpp file and create a cpp file containing a main method that instantiates a CSRMatrix<T> object and populates it. This sparse matrix should have a number of non zeros smaller than the number of rows times the number of columns, i.e., a matrix that has some zeros in it. *Also, some of those zeros **should** be on the diagonal*. The main method should then call the computeTrace method on that object and print the resulting trace value.

Question C.9

Implement a copy constructor for the CSRMatrix class, that takes an existing CSRMatrix<T> object as input and then instantiates and copies the values into the existing object. This copy constructor should not be just produce a "shallow" copy of input (i.e., if the input is changed at some point, the values of the new object should not change).

As part of your work on linear solvers, you have decided to implement an incomplete LU factorisation, which computes matrices such that for a given matrix **A ≈ LU,** where L and U are lower and upper triangular matrices, respectively. Writing an ilu(0) method can be difficult however, so you decide to just implement part of an ilu(0) method, given by the following pseudo-code:

If **A** is a matrix of size nxn:

```
for $i = 2$ to $n$
    for $k = 1$ to $i-1$
        $A[i][k] = A[i][k] / A[k][k]$
```

Implement this algorithm for the CSRMatrix class, by creating a method which takes no arguments and modifies the values of the matrix **A** "in-place", i.e., by overwriting the values of **A**. Do not at any point convert **A** to a dense matrix or use an amount of memory equivalent to **A** stored densely, you must operate on the sparse matrix.

Modify your CSRMatrix.cpp file and create a cpp file containing a main method that instantiates a CSRMatrix<T> object and populates it. This sparse matrix should have a number of non zeros smaller than the number of rows times the number of columns, i.e., a matrix that has some zeros in it. *Also, **none** of those zeros should be on the diagonal*. The main method should then copy the CSRMatrix using your new copy constructor, and then compute an ilu(0) on the new copy. Print both the original CSRMatrix and the newly decomposed matrix.