Lab Books for 700106 & 700120

Contents

[Real Time Graphics (700106) Labs 2](#_Toc120614210)

[Week 1 – Lab 1 2](#_Toc120614211)

[Week 2 – Lab 2 9](#_Toc120614212)

[Week 3 – Lab 3 15](#_Toc120614213)

[Week 4 – Lab 4 22](#_Toc120614214)

[Week 5 – Lab 5 32](#_Toc120614215)

[C++ Programming & Design (700120) Labs 40](#_Toc120614216)

[Week 1 – Lab A 40](#_Toc120614217)

[Week 2 – Lab B 48](#_Toc120614218)

[Week 3 – Lab C 58](#_Toc120614219)

[Week 4 – Lab D 66](#_Toc120614220)

[Week 5 – Lab E 73](#_Toc120614221)

[Week 6 – Lab F 87](#_Toc120614222)

[Week 7 – Lab G 95](#_Toc120614223)

[Week 8 – Lab H 101](#_Toc120614224)

[Final Lab 108](#_Toc120614225)

[Graphics (700106) 108](#_Toc120614226)

[Design (700120) 108](#_Toc120614227)

# Real Time Graphics (700106) Labs

## Week 1 – Lab 1

Date: 28th Sep 2022

### Q1. Exercise 0

#### Question:

Change the “Eye” view so that you can see the top of the animated cube

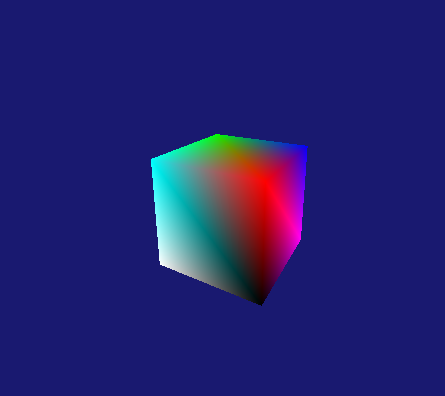
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was very straight-forward just had to change the y position of the “Eye” XMVECTOR

#### Metadata:

Change of view

#### Further information:

n/a

### Q2. Exercise 1

#### Question:

Modify the vertex list in indices[] or modify the parameters in the DrawIndexed( ) to draw:

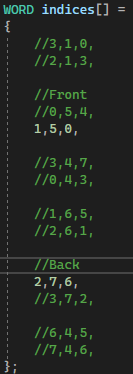
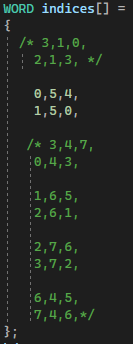
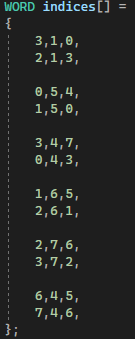
1. two triangles

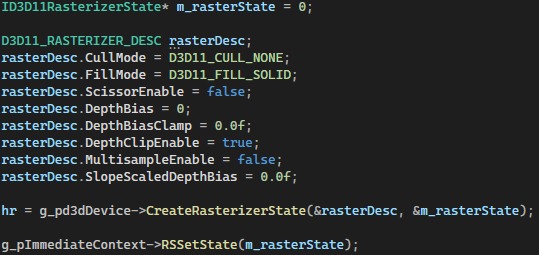
2. one face of the cube

3. the four walls of the cube

#### Solution:

(1) (2) (3)

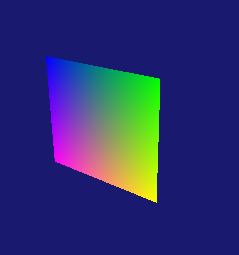
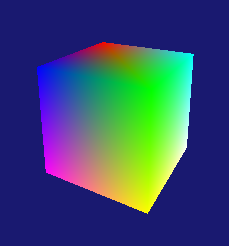


#### Test data:

n/a

#### Sample output:

1. 2. 3.

#### Reflection:

The rasterizer seems complicated at the moment but everything else was fairly easy

#### Metadata:

Faces

#### Further information:

n/a

### Q3. Exercise 2

#### Question:

Draw the cube as a wireframe

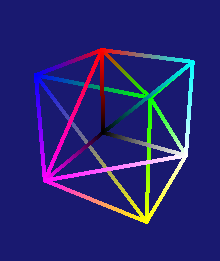
#### Solution:



#### Test data:

n/a

### Sample output:



#### Reflection:

This was really easy to do. I only had to change half a line of code

#### Metadata:

Wireframe

#### Further information:

n/a

### Q4. Exercise 3

#### Question:

Modify the parameter in “IASetPrimitiveTopology( )” and “indices[]” to draw:

1. A list of points corresponding to the cube’s eight vertices.

2. The 12 edges of the cube (not as a wireframe triangle mesh).

#### Solution:

1.



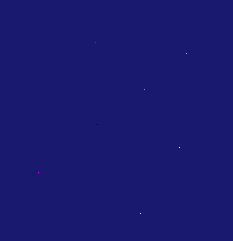
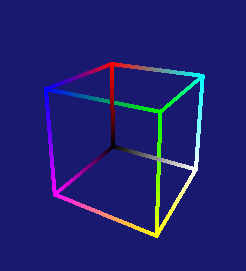


#### Test data:

n/a

#### Sample output:

1. 2.

#### Reflection:

It took a while to figure out part 2, I was drawing too many lines for what was needed

#### Metadata:

Topology

#### Further information:

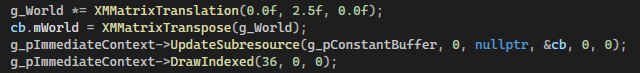
n/a

### Q5. Exercise 4

#### Question:

Draw two wireframe cubes

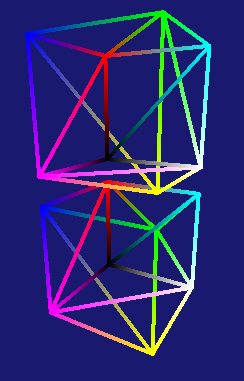
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

Figuring out where I needed to put the code was the only hard part, but at least now I know.

#### Metadata:

Twice the cubes

#### Further information:

n/a

### Q6. Exercise 5

#### Question:

Draw the cube as triangle strips by setting primitive topology as D3D11\_PRIMITIVE\_TOPOLOGY\_TRIANGLESTRIP

#### Solution:

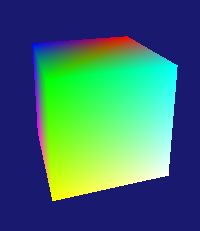




#### Test data:

n/a

#### Sample output:



#### Reflection:

This took me way too long and I still couldn’t get it perfect. I understand triangle-strip conceptually, I just can’t visualise it clearly enough in my head to implement it perfectly.

#### Metadata:

Triangle strip

#### Further information:

How can I better visualise it to implement it correctly?

## Week 2 – Lab 2

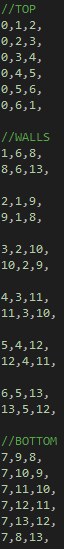
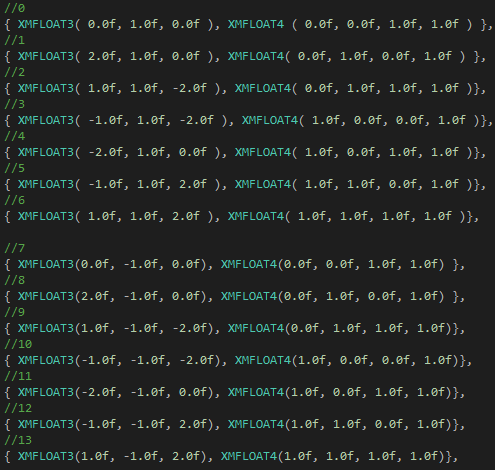
Date: 05/10/2022

### Q1. Exercise 1

#### Question:

Modify the vertex list and indices to create a hexagonal cylinder

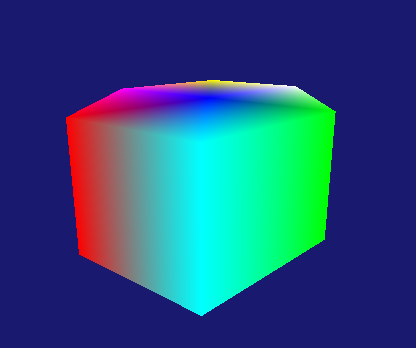
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This took way longer the it needed because I missed a single line. I forgot to update the size of the vertex list. 

#### Metadata:

Hex

#### Further information:

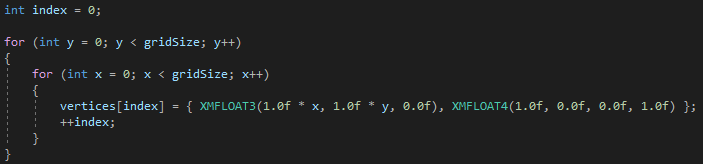
n/a

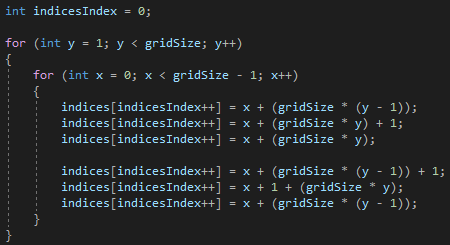
### Q2. Exercise 2

#### Question:

Modify the cube vertex list in the sample to specify a flat 3D grid and display it as a wireframe

#### Solution:

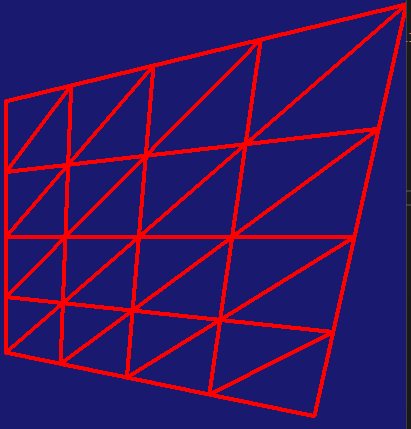




#### Test data:

n/a

#### Sample output:



#### Reflection:

The procedural indices took forever to figure out how to do, I was over engineering the solution needed.

#### Metadata:

Procedural

#### Further information:

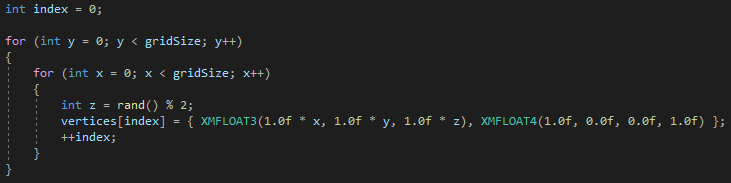
n/a

### Q3. Exercise 3

#### Question:

Specify different heights at different grid points for the 3D grid you created in Exercise 6 to create a terrain triangle mesh

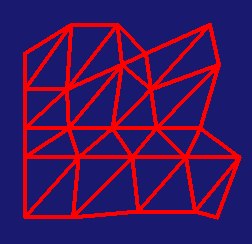
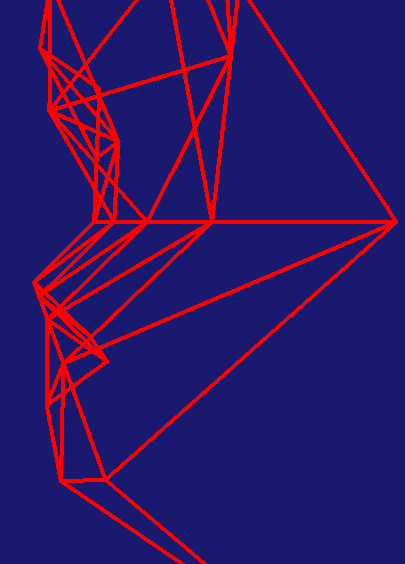
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was pretty straight-forward once I got the procedural indices to work. It’s a bit hard to get a good angle to show off that it has got an z axis. I probably should have done a wider range because at the moment the z axis is only ever 0.0f or 1.0f.

#### Metadata:

Terrain

#### Further information:

n/a

### Q4. Exercise 4

#### Question:

Use e “GeometryGenerator.cpp” , “GeometryGenerator.h” and “ShapesApp.cpp” to draw a sphere, cylinder and plane.

#### Solution:

Text

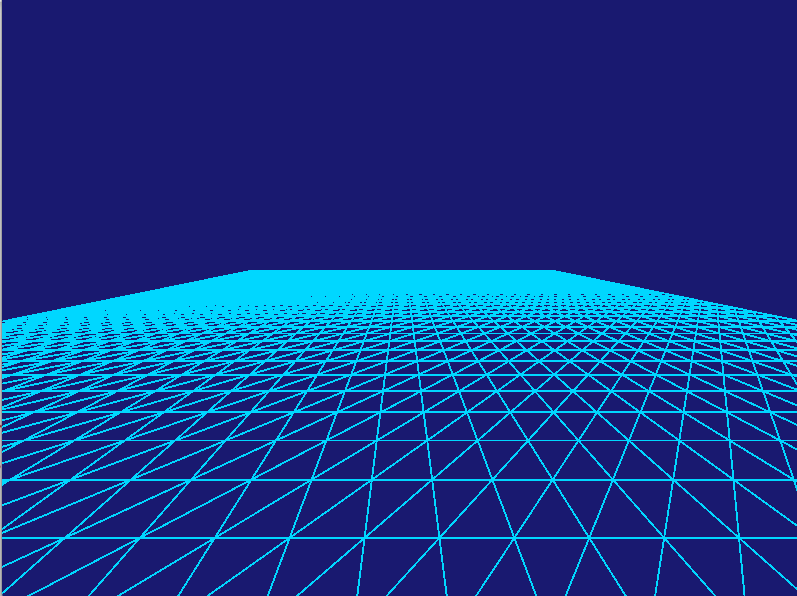
Description automatically generated 

These are pretty much the same for all the shapes the only difference is the keyword changes e.g. “grid”.

#### Test data:

n/a

#### Sample output:

 Shape

Description automatically generated A picture containing text

Description automatically generated

#### Reflection:

This was pretty straight forward once I had spotted the simple mistake I had made. I’m noy sure how you would draw all of them as once like in the “ShapesApp.cpp” example.

#### Metadata:

Geometry generator

#### Further information:

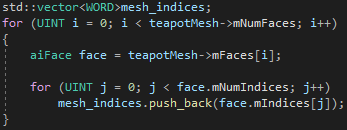
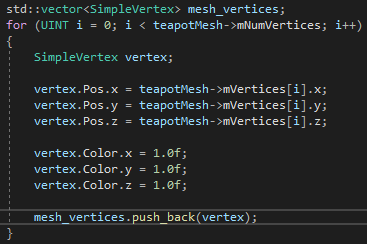
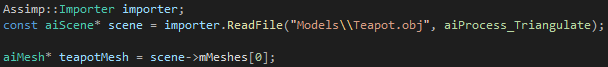
n/a

### Q5. Exercise 5 Loading graphics assets using AssImp

#### Question:

Use AssImp to draw the teapot model given

#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

I’ve done everything right; the code successfully compiles but I keep getting the error shown above. Neither the lecturers or my class mates can find anything wrong with what I’ve done, and the error appears not matter what PC I’m on or what obj file I use. So, I just have to move on to the next tasks.

#### Metadata:

Assimp

#### Further information:

If why this is happening can be figured out and fixed then I will finish this lab properly

## Week 3 – Lab 3

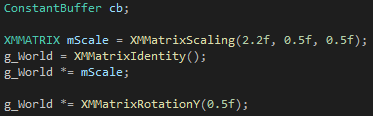
Date: 11/10/2022

### Q1. Exercise 1

#### Question:

Transform the cube using scaling transformation

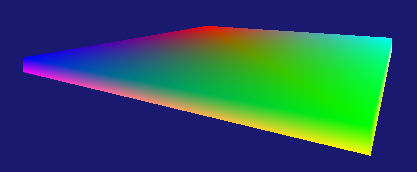
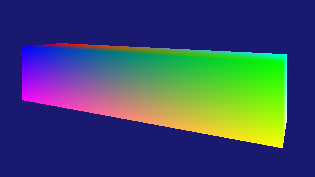
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

These were really easy to do, the only challenge was getting the proportions right to make shapes like the examples

#### Metadata:

Scaling

#### Further information:

n/a

### Q2. Exercise 2

#### Question:

Perform scaling, translation, and rotation transformation to achieve the following effects:

1. A cube rotates by a vertical rotation axis
2. Two cubes rotate by two different rotational axes with different rotational speeds respectively

#### Solution:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

#### Test data:

n/a

#### Sample output:

1. 2.

Shape

Description automatically generated A picture containing icon

Description automatically generated

#### Reflection:

This was fairly simple to do once I realised that I needed to create a depth stencil to make the cubes go behind the pillar properly.

#### Metadata:

Cube around a pillar

#### Further information:

Transformation order is Scale the rotation then translation. However, different orders can be used for different effects

### Q3. Exercise 3

#### Question:

scale the small rotating cube into a long-thin stick and rote the stick by a rotation axis such that the stick is always tangent to the rotation path.

#### Solution:

Text

Description automatically generated

#### Test data:

n/a

#### Sample output:

Background pattern

Description automatically generated

#### Reflection:

This is the exact same as the last exercise except the scaling is different. Have I misunderstood what one of the exercises is asking for?

#### Metadata:

Tangent rotation

#### Further information:

n/a

### Q4. Exercise 4

#### Question:

Scale the cube into different sizes corresponding to the Sun, the Earth and the Moon respectively and then combine a set of rotation and translation transformations to animate a simple solar system

#### Solution:

Text

Description automatically generatedA screenshot of a computer

Description automatically generated with medium confidenceText

Description automatically generated

#### Test data:

n/a

#### Sample output:

Shape

Description automatically generated

#### Reflection:

The sun and the earth were easy to do but the moon I can’t quite figure out. I can get it to be the right size and have its own spin, but I can’t get it to orbit the earth. I’m not sure how to make the origin point for the moons orbit be the centre of the earth cube.

My second attempt at getting the moon right looks like its work but its hard to tell if it actually is.

#### Metadata:

Solar system

#### Further information:

n/a

### Q5. Exercise 5

#### Question:

Manually set the g\_View and g\_Projection and observe if you get the same effect

#### Solution:

Text

Description automatically generated

Text, chat or text message

Description automatically generated

#### Test data:

n/a

#### Sample output:

Shape, square

Description automatically generated

#### Reflection:

I seem to get same effect but this could be because I copied the matrices that get produce by the XMMatrixLookAtLH() and XMMatrixPerspectiveFovLH() and hardcoded them.

#### Metadata:

Hardcoded matrices

#### Further information:

n/a

## Week 4 – Lab 4

Date: 18/10/2022

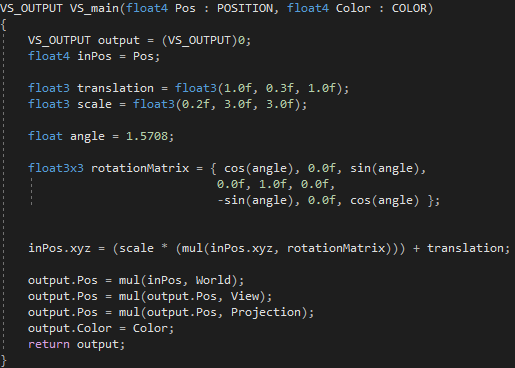
### Q1. Create your own vertex shaders

#### Question:

1. Create a new vertex shader in “Tutorial04.fx” this vertex shader should translate by (1.0f, 0.3f, 1.0f) , scale by (0.2f, 3.0f, 3.0f) and rotate by the y-axis.
2. Create a new HLSL file and copy your new vertex shader into it then proceed to set it properties correctly.

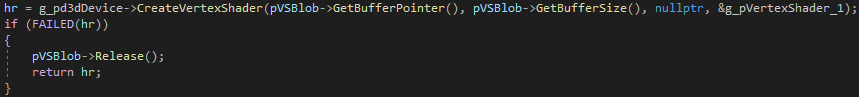
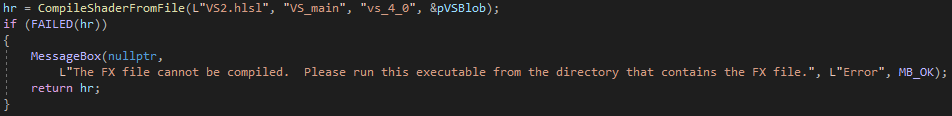
#### Solution:

1.



2.



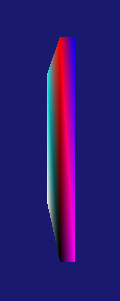


#### Test data:

n/a

#### Sample output:

1. 2.

 A picture containing background pattern

Description automatically generated

#### Reflection:

This took a while to do, I don’t really understand HLSL very well so I need to practice more. Switching from an .fx file to a .hlsl file didn’t seem to be much different.

#### Metadata:

HLSL

#### Further information:

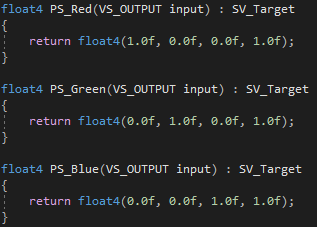
n/a

### Q2. Create your own pixel shaders

#### Question:

Create pixel shaders to make 3 different coloured cubes

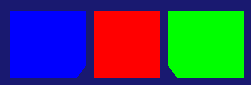
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was pretty simple to do. There is a lot of repetitive code to compile and create each pixel shader though, I’m sure that could improve on that by making a vertex shader and pixel shader class. I’m not sure why a chunk of the blue and green cube is missing, if I set the translations so each cube is right next to each other then it doesn’t happen

#### Metadata:

Pixel shaders

#### Further information:

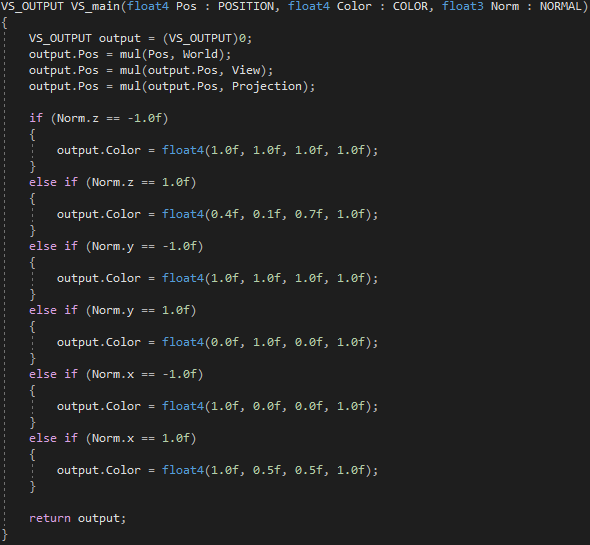
n/a

### Q3. Cornell box in vertex shaders

#### Question:

Create Cornell box by using THREE different vertex shaders

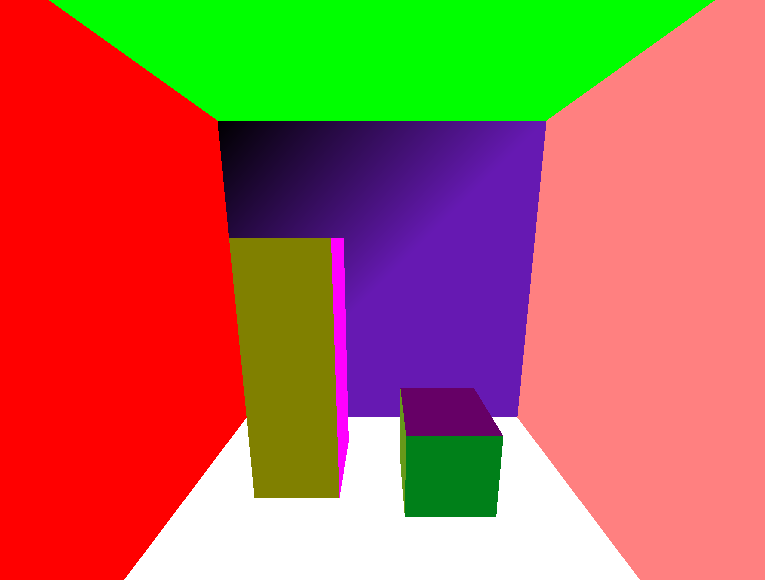
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

It doesn’t look exactly like the example we were given but its close enough I think. For some reason the vertex shader wont colour the last vertex and I’m not sure why.

#### Metadata:

Cornell

#### Further information:

n/a

### Q4. Define Model-View-Projection in vertex shader (optional)

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

As this is optional I will skip it for now as I am currently behind in the labs

#### Metadata:

#### Further information:

### Q5. Vertex shader point cloud (Optional)

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

As this is optional I will skip it for now as I am currently behind in the labs

#### Metadata:

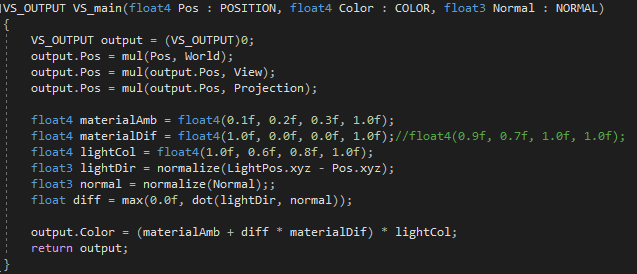
#### Further information:

### Q6. Per-vertex diffuse lighting

#### Question:

Create diffuse lighting in your vertex shader. Then modify the material reflection coefficient so that only red light is reflected

#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was pretty straight-forward to do there isn’t much to comment on. There was quite a lot of set up work to get this to work and I’m not sure if I really understand the depth-stencil texture and depth-stencil view.

#### Metadata:

Light

#### Further information:

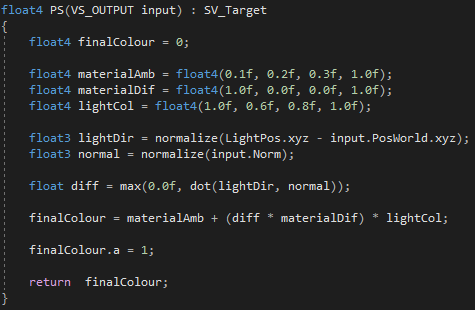
n/a

### Q7. Per-pixel diffuse lighting

#### Question:

Create diffuse lighting in the pixel shader

#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

Finally got this done it turns out I wasn’t passing the light position to the pixel shader. Lighting and Shader programming is still really confusing to me though

#### Metadata:

Pixel lighting

#### Further information:

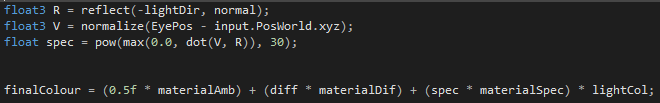
n/a

### Q8. Per-pixel specular lighting

#### Question:

Create specular lighting in the pixel shader

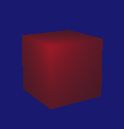
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was fairly easy to do now that I have my pixel shader lighting working. I need to remember that with specular light that on this line:  the 30 represents shininess of the surface 1 being most shiny 200 being the least

#### Metadata:

Specular Pixel

#### Further information:

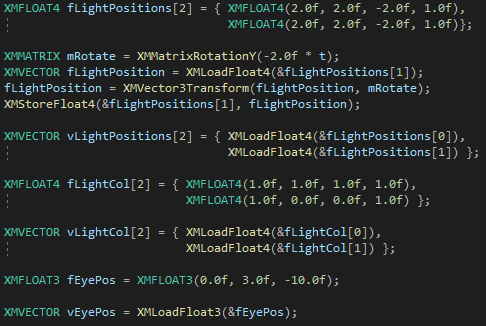
n/a

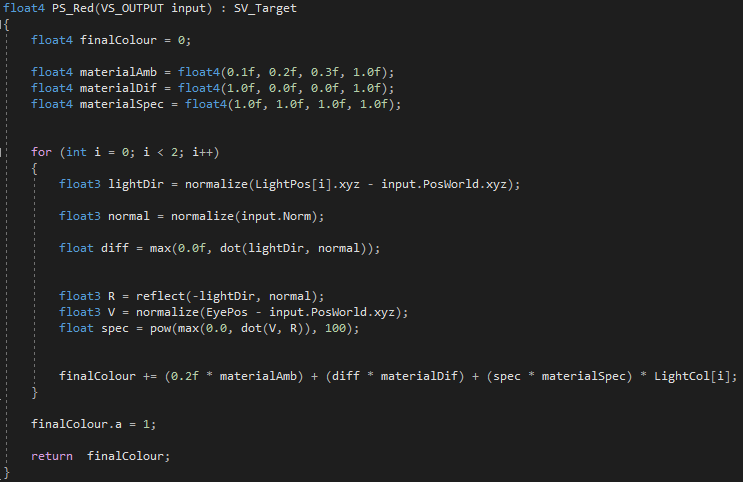
### Q9. Multiple materials and light sources

#### Question:

Draw three cube objects with three different surface material properties. Illuminate these cubes with two light sources: one is a static white light, the other is a red light rotating dynamically by the y-axis.

#### Solution:

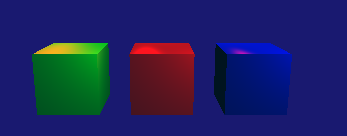




#### Test data:

n/a

#### Sample output:



#### Reflection:

I’m not completely confident that I have done this correctly. Based on what I know and by looking at tutorial06 I think I have successfully done it.

#### Metadata:

n/a

#### Further information:

Can you confirm if I have implemented this correctly? and if I haven’t can you point out where I went wrong please. For some reason I’m struggling to understand lighting fully.

## Week 5 – Lab 5

Date: 26/10/2022

### Q1. A wooden box

#### Question:

Give your cube a wood texture

#### Solution:

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generated

#### Test data:

n/a

#### Sample output:

A picture containing container, wooden, box, building material

Description automatically generated

#### Reflection:

This was pretty straight-forward the only issue I had was that the layout creation was failing but I eventually found out I had just misspelled one of the matching variables in the vertex shader.

#### Metadata:

Wooden cube

#### Further information:

n/a

### Q2. Texture wrapping mode

#### Question:

Create a texture-mapped cube using the coin texture

#### Solution:

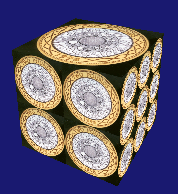
Graphical user interface, text

Description automatically generated

#### Test data:

n/a

#### Sample output:



#### Reflection:

This was simple but I don’t really understand the other types of texture addresses or when I might want to use them

#### Metadata:

Texture addresses

#### Further information:

n/a

### Q3. Mipmapping

#### Question:

Create mipmaps of a loaded texture and use HLSL sampleLevel( ) to map the cube with different level of mipmaps

#### Solution:

Text

Description automatically generated

#### Test data:

n/a

#### Sample output:

A picture containing text, envelope

Description automatically generated

#### Reflection:

This took way too long to do; I couldn’t get it to work but then after trying it in tutorial 7 and comparing the code I realised I had not set somethings in the sampler description properly.

#### Metadata:

Mipmaps

#### Further information:

n/a

### Q4. Texture filtering techniques

#### Question:

Scale the cube along the view direction to create a long rectangular object. Using different filtering techniques to deal with the minification and magnification issues on texture mapping and observe how visual quality of the rendered image is being changed.

#### Solution:

1. 
2. 
3. 
4. 

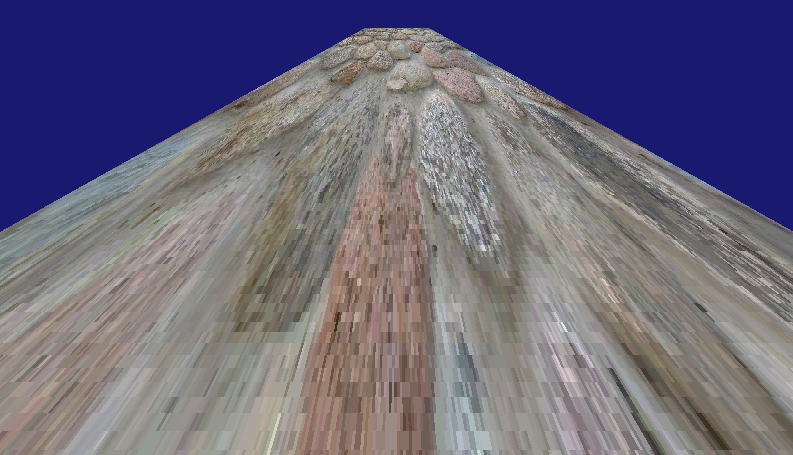
#### Test data:

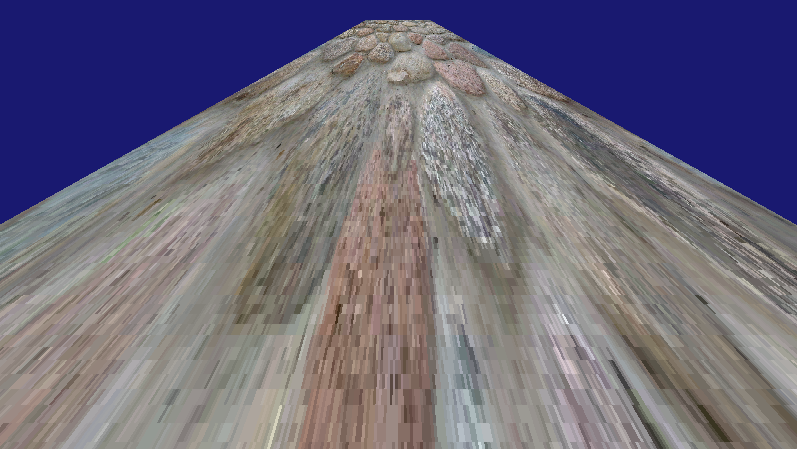
n/a

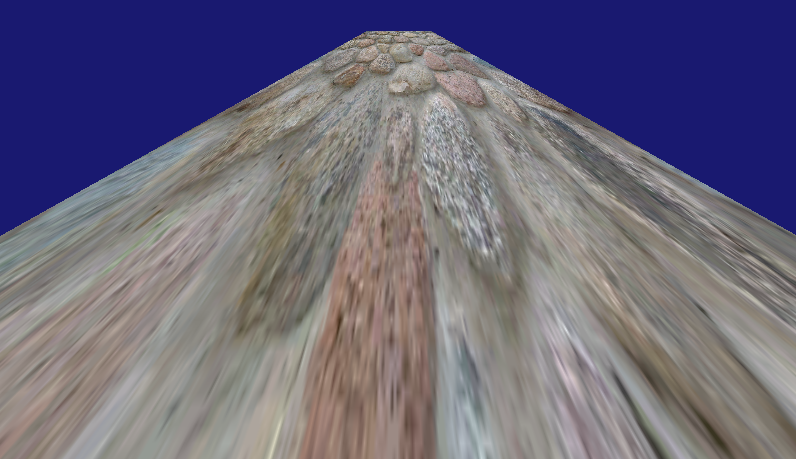
#### Sample output:

A sand dune with a blue sky

Description automatically generated with low confidence







#### Reflection:

2 and 3 seem to reduce the quality of the texture and make it much more grainy. 1 and 4 seem to be pretty similar, 1 might be a little bit sharper. I’m not sure on the use cases for each filter I guess it just depends on the effect you are trying to achieve

#### Metadata:

Filters

#### Further information:

n/a

### Q5. Multiple texturing

#### Question:

Create a cube that uses the tile with the coin texture layered on top.

#### Solution:

Text

Description automatically generated

#### Test data:

n/a

#### Sample output:

A picture containing calendar

Description automatically generated

#### Reflection:

This took a lot longer than it should of. I was trying to test the alpha of the texture by looking in the 4th float of a float4 but since the texture is 2-D the alpha is actually stored in the 3rd float of a float4. I think that’s right. Please correct me if I am wrong because I’m not full sure how I managed to achieve the desired effect.

#### Metadata:

Multiple Textures

#### Further information:

n/a

# C++ Programming & Design (700120) Labs

## Week 1 – Lab A

Date: 27th Sep 2022

### Q1. Hello World

#### Question:

Locate the Solution Explorer within Visual Studio and select the HelloWorld project. Right click on this project and select Build. This should compile and link the project. Now run the HelloWorld program.

Change between Debug and Release mode. Compile again and rerun the program.

#### Solution:

#include <iostream>

int main(int, char\*\*) {

std::cout << "Hello World" << std::endl;

return 0;

}

#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

This is programming 101

#### Metadata:

Hello World

#### Further information:

What are purpose are the parameters in main?

### Q2. Console Window

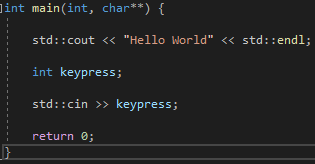
#### Question:

Delay the termination of the program by adding the following two lines to the end of your code:

int keypress;

std::cin >> keypress;

#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This just stops the program from terminating as soon as the program what finished running.

#### Metadata:

Delay termination

#### Further information:

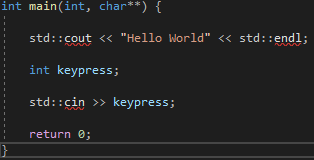
Is this something that’s only really done for educational purposes?

### Q3. Includes

#### Question:

Remove “#include <iostream>” then compile, what is the effect>

#### Solution:



#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

cin and cout can’t be used because the program doesn’t have access to the iostream

#### Metadata:

includes

#### Further information:

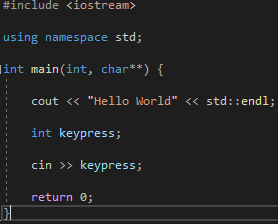
n/a

### Q4. Namespace

#### Question:

Add “using namespace std;” and remove all “std::”

#### Solution:



#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

I thought that visual studio would make you get rid of the “std::” to compile if you as using the std namespace but apparently not

#### Metadata:

Namespace std

#### Further information:

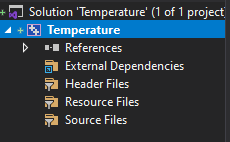
n/a

### Q5. Create a new project

#### Question:

Create a new Visual C++ Empty project called “Temperature”

#### Solution:



#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

Very straight-forward, nothing to comment on

#### Metadata:

New project

#### Further information:

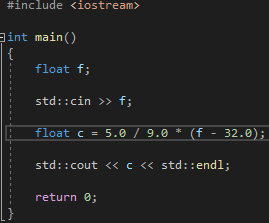
n/a

### Q6. Temperature

#### Question:

Create a new cpp file within the temperature project and write a program to input a Fahrenheit measurement, convert it and output a Celsius value.

#### Solution:



#### Test data:

Input: 32 output: 0

Input: 33 output: 0.555556

#### Sample output:





#### Reflection:

Need to remember that is I want a float answer then the numbers in the calculation should be floats as well.

#### Metadata:

Conversion

#### Further information:

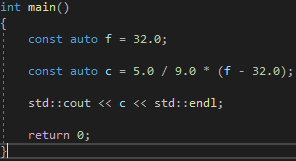
n/a

### Q7. Auto, const and casting

#### Question:

Now rewrite your temperature example using the auto keyword, constants and explicit casting

#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

I would say that this takes more thought to read than the previous implementation but this defiantly uses less lines of code, but also has less functionality.

#### Metadata:

keywords

#### Further information:

What would be some use cases for the auto keyword?

### Q8. Static assert

#### Question:

Create a new project call sizeOf. Select a different architecture (e.g. x86 or x64) to see if you can make the assert fail.

#### Solution:



#### Test data:

Switch between x86 and x64 architecture

#### Sample output:

n/a

#### Reflection:

The assert doesn’t work on the x64 architecture. I’m guessing we will find out why when the lectures start.

#### Metadata:

asserts

#### Further information:

n/a

## Week 2 – Lab B

Date: 04/10/2022

### Q1. Timing

#### Question:

Run the given application on your pc and examine the results. Try increasing the loop limit and architecture.

#### Solution:

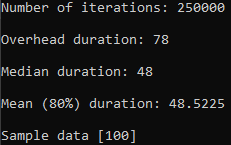
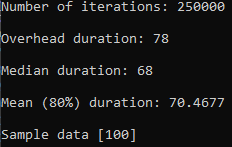


#### Test data:

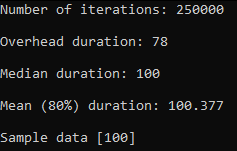
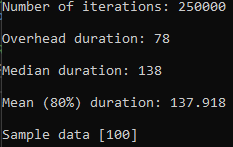
Loop limit = 20 Loop limit = 40

#### Sample output:

Loop limit: 20 Architecture: x64 Loop limit: 20 Architecture: x86

Loop limit: 40 Architecture: x64 Loop limit: 40 Architecture: x86

#### Reflection:

There is an increase in median duration when on a x86 architecture.

When increasing the loop limit the mean duration is a little over double the time. The only explanation I can think of is since there is double the amount of loops then there is more work to do and more opportunity of the branching to be wrong.

#### Metadata:

Timings

#### Further information:

Remember that timing in debug is pointless

### Q2. Timing own code

#### Question:

Replace the payload with some of your own code

#### Solution:

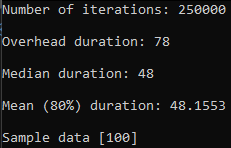
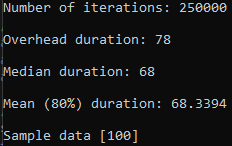


#### Test data:

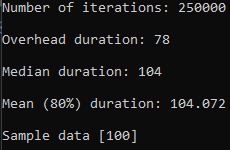
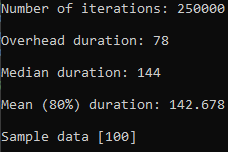
Loop limit = 20 Loop limit = 40

#### Sample output:

Target: 20 Architecture: x64 Target: 20 Architecture: x86

Target: 40 Architecture: x64 Target: 40 Architecture: x86

#### Reflection:

My timing code didn’t seem to have much different times from the given example even though more operations are happening in my for loop.

#### Metadata:

Own code

#### Further information:

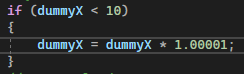
n/a

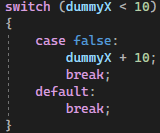
### Q3. Conditionals

#### Question:

Add each of the conditional statements in turn to the payload to try and identify any performance differences.

#### Solution:





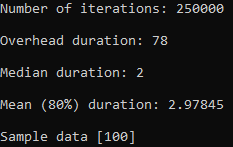
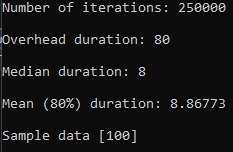


#### Test data:

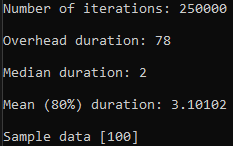
n/a

#### Sample output:

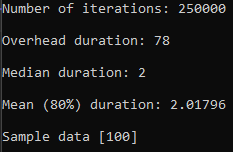
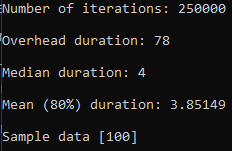
If x64 – If x86 –

Switch x64 – Switch x86 –

?: x64 – ?: x86 –

#### Reflection:

All of the different conditionals seem pretty similar, they all do worse on x86 architecture as expected. It seems the ?: does a little bit better than if on a x86 architecture though

#### Metadata:

Conditions

#### Further information:

n/a

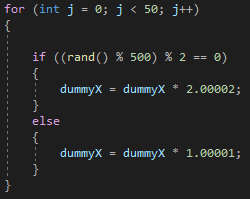
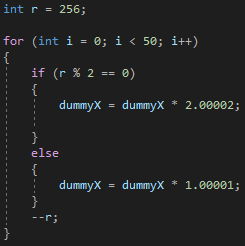
### Q4. Branch prediction

#### Question:

Add a piece of code to the payload that demonstrates when branch prediction is working well and when branch prediction is failing

#### Solution:

Fail Pass

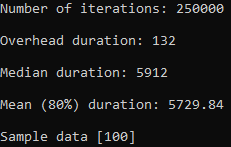
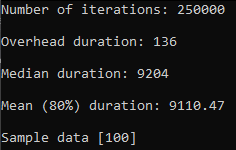
 

#### Test data:

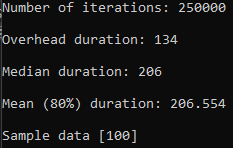
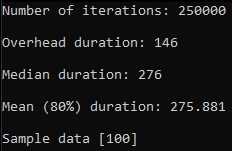
n/a

#### Sample output:

Fail x64 x86

Pass x64 x86

#### Reflection:

I am not 100% sure that I have done this right but the “fail” example defiantly does worse in terms for speed. The branch predictor should only have a 50-50 chance on predicting right in the fail example because the conditions are reliant on a random number.

#### Metadata:

Failure

#### Further information:

Check with Warren to see if I have done this correctly

### Q5. Exiting a nested loop

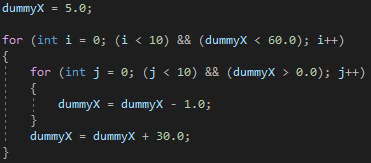
#### Question:

Add each option for exiting a nested loop to the payload section of the timing code and determine if there are any performance differences between each approach

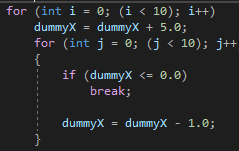
1. Two conditions in each conditional section of the loops. One for the loop control and the other as the exit condition
2. An additional if statement immediately following the inner loop to catch and propagate a break statement
3. A goto statement in the inner loop
4. A lambda function

#### Solution:

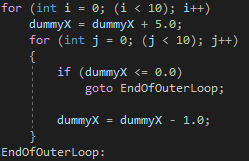
1.



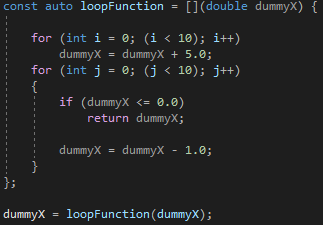
2.



3.



4.

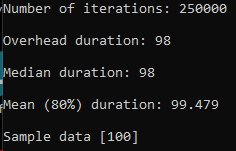
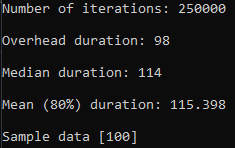


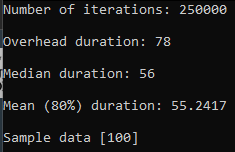
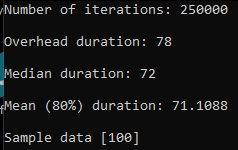
#### Test data:

n/a

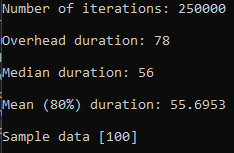
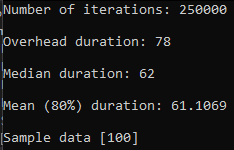
#### Sample output:

1. x64 x86

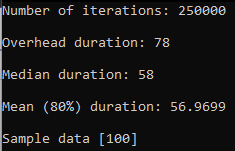
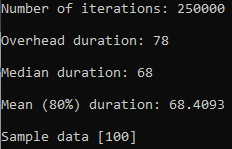


2. x64 x86  
 

3. x64 x86

4. x64 x86

#### Reflection:

I am not sure I have implemented all of these correctly but based on my results all of the methods of exiting a loop are fairly similar in speed except for method 1 (Two conditions in each conditional section of the loops. One for the loop control and the other as the exit condition). When running the tests in x86 architecture they all run slower than the x64 as expected.

#### Metadata:

Nested

#### Further information:

They are supposed to be all pretty equal method 1 is wrong because of the ovehead

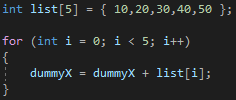
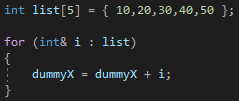
### Q6. Range based loop

#### Question:

Compare the performance of standard loops and range based loops

#### Solution:

Standard Range-Based

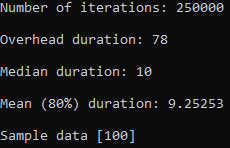


#### Test data:

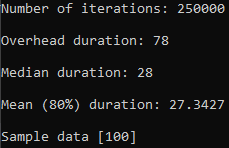
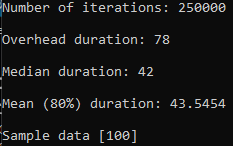
n/a

#### Sample output:

Standard x64 x86

Range-Based x64 x86

#### Reflection:

Ranged-based is significantly slower both in x64 and x86. I was expecting it to be quicker. Maybe I’ve implemented it wrong

#### Metadata:

Range-Based

#### Further information:

Are my results as expected? If so why is it so much slower? **They are actually pretty equal in speed**

### Q7. Architecture

#### Question:

Run the previous experiments in both x64 and x86 architechture

#### Solution:

n/a

#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

I’ve done this as part of each question and commented on the architecture differences in each

#### Metadata:

X86

#### Further information:

n/a

## Week 3 – Lab C

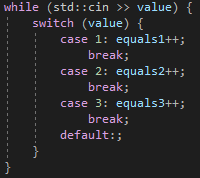
Date: 11/10/2022

### Q1. Debugging

#### Question:

Determine what the problem is with the program and suggest a solution to make the program execute correctly.

#### Solution:



#### Test data:

#### Sample output:

#### Reflection:

The problem was that the switch didn’t have a break at the end of each case causing the code to fall through. So, if case 1 was triggered so would case 2 and 3. The fix is simply adding in the breaks for each case

#### Metadata:

Switch error

#### Further information:

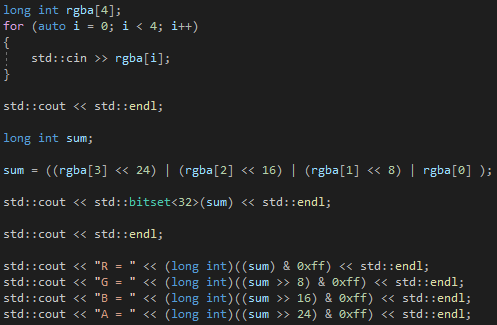
n/a

### Q2. Bitwise

#### Question:

Write a program to read four separate 32-bit integers (red, green, blue, and alpha) and encode them into a single 32-bit value. Output this 32-bit value. Verify that the results are correct by taking the 32-bit value and extracting and outputting the separate integers.

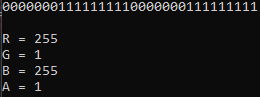
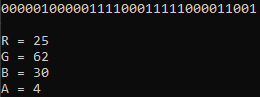
#### Solution:



#### Test data:

#### Sample output:

#### Reflection:

This took a while and I had to get quite a bit of help but I think I understand bitwise a bit better now. I should still do some more research into bitwise stuff though.

#### Metadata:

Bitwise encoding

#### Further information:

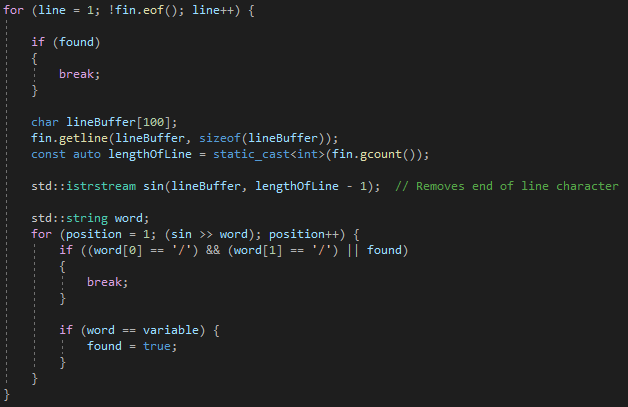
n/a

### Q3. Parsing

#### Question:

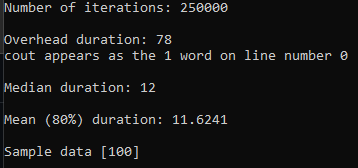
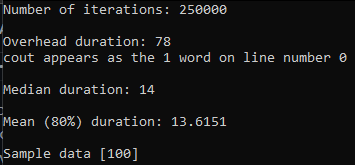
Modify the program to make the loop structures more efficient and easier to maintain. Time the code to determine if you have made to code more efficient.

#### Solution:

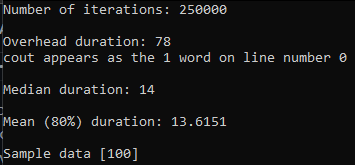


#### Test data:

Release x64 Original Mine



#### Sample output:



#### Reflection:

My solution is ever so slightly faster however when I was doing the timing the times for the original seemed to vary wildly so I’m not 100% sure if it is more efficient. In terms of maintainability, I’d say the 2 solutions are equal and I’m not sure how you could improve maintainability.

#### Metadata:

Parsing improvement

#### Further information:

n/a

### Q4. Quadratic

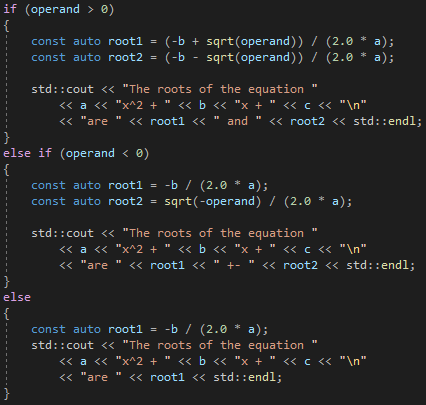
#### Question:

Modify the quadratic program to:

1. Handle equal or imaginary roots
2. Output the roots to only 3 decimal places

Time the new code and try to improve efficiency

#### Solution:





#### Test data:

#### Sample output:

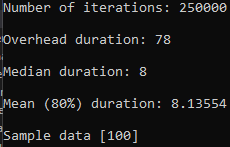


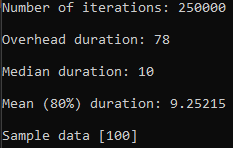




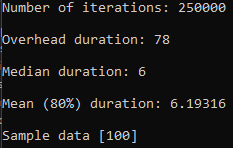
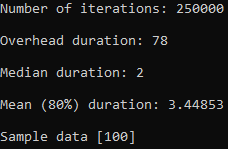
Release x64

Real & Diff

 Text

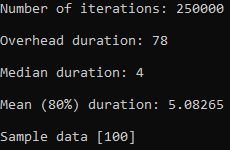
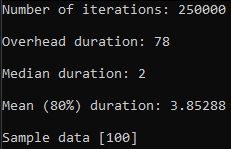
Description automatically generated 

Imaginary

  Text

Description automatically generated

Real Same

  Text

Description automatically generated

#### Reflection:

This was really hard to do purely because I don’t remember anything about quadratic equations. When I came to the try to improve efficiency part I couldn’t think of any way to get any improvement. I need to use a conditional for the program and the other types of conditionals aren’t appropriate. I think I remember seeing Warren do the precision differently in a lecture but I couldn’t find it or remember it, I think % was used somehow.

#### Metadata:

Quadratic equations

#### Further information:

I think I saw Warren do precision another way so I need to ask him how he does it

I noticed that the overhead massively increases if my laptop is unplugged. It goes from ~80 when plugged in to ~200 when unplugged.

### Q5. Assembly

#### Question:

Open and step through the assembly code from the parsing exercise. Do this in debug mode.

#### Solution:

Text

Description automatically generated

Text

Description automatically generated

#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

I’ve gone through the assembly and I understand what is happening for the most part but I think that’s only due to the fact that I know what’s supposed to be happening in the code, if I was just given the assembly code with no other context I don’t think I’d be able to tell you what the related piece of code was doing.

After going through the assembly code, I noticed there were a few instructions that I didn’t recognise so I’ve found out what they mean and put them below:

nop = No operation

inc = Increment (+1)

movsx/movsxd = Move with Sign-Extension

movzx = Move with Zero-Extend

test = Preforms a bitwise and

dec = Decrement (-1)

xor = Perform a bitwise exclusive or

#### Metadata:

Parsing Assembly

#### Further information:



Why is there a d on the end of the R9 and R8 register?

### Q6. Optimizer

#### Question:

In release mode disassemble the code you wrote for the quadratic exercise in the timing code.

#### Solution:

Text

Description automatically generated

#### Test data:

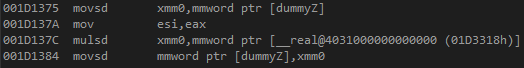
n/a

#### Sample output:

n/a

#### Reflection:

The optimiser seems to have completely thrown out my code and seems to be figuring out the answers in the following section ready to print later:



I don’t really understand what’s happening or why. I though since the roots and operand are need to alter the dummy values before they get printed that the optimiser would leave my code alone.

mulsd = Multiply Scalar Double-Precision Floating-Point Value

cmovg =  conditional move if greater check the state of ZF, SF AND OF. If ZF=0 AND SF=OF then condition is satisfied, otherwise it will be skipped.

cmp = subtracts one operand from the other for comparing whether the operands are equal or not

#### Metadata:

Optimiser

#### Further information:

Is this what was expected or have I done something wrong. **The code has note been optimised out it is all below the payload the disassembler just gave up doing it line by line.**

## Week 4 – Lab D

Date: 25/10/2022

### Q1. Object Parser

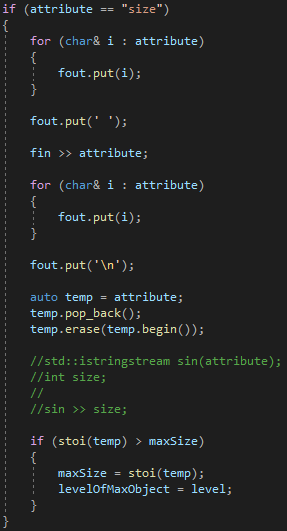
#### Question:

Complete the object parser code. It must be able to copy data to an output file and identify the object with the largest size.

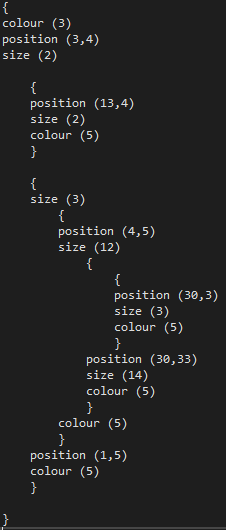
#### Solution:

Text

Description automatically generatedText

Description automatically generated

#### Test data:



#### Sample output:

Text

Description automatically generated

#### Reflection:

This wasn’t too difficult the most difficult part was converting a size string into an int. Warren suggested using istringstream but I don’t think I was using it right because it always gave a size of 0.

Note: I know I could have just used streaming operators for writing but I wanted to practice using range-based for loops.

#### Metadata:

Parsing Objects

#### Further information:

What am I doing wrong with the istringstream?

### Q2. Object Parser (Recursive)

#### Question:

Create a recursive solution to the previous task. No loop structures allowed

#### Solution:

Text

Description automatically generated

#### Test data:

Same as previous task

#### Sample output:

Same as previous task

#### Reflection:

This was pretty straight forward, I probably have too many parameters so it’s a bit messy and has a lot of room for errors to be made.

#### Metadata:

[Recursion](#_Q2._Object_Parser)

#### Further information:

n/a

### Q3. Object Parser Comparison

#### Question:

Write some timing code a use it to compare the speed of object parsing with loop structures and with recursion.

#### Solution:



#### Test data:

n/a

#### Sample output:

|  |  |
| --- | --- |
| **Loop structures Release x64** | **Recursion Release x64** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| *Average Time = 106560 nanoseconds* | *Average Time = 112300 nanoseconds* |

#### Reflection:

The timings seem pretty similar loop structures do come out to be a bit faster in my timings but this is probably just luck of the draw since I just took 5 random timings for each solution. I probably should have looped through multiple iterations and gotten the average like that.

#### Metadata:

Comparison

#### Further information:

For the loop structures I changed to way things a written to the output.txt so they matched across both solutions

### Q4. Tuples

#### Question:

Select one of your object parsers and wrap the code within a new function. This new function has a single parameter, the file name, and returns a tuple consisting of the size of the largest object and the level of the largest object

#### Solution:

Text

Description automatically generatedText

Description automatically generated

#### Test data:

Sample.txt

#### Sample output:



#### Reflection:

This was pretty simply to do, I cut out most of the code since we aren’t writing to the out put file for this question.

#### Metadata:

Tuple

#### Further information:

n/a

### Q5. Span and Arrays

#### Question:

1. Create a version of the code that uses the array template, which wraps a vanilla C array within a C++11 template.
2. Create a version of this function that uses the C++20 span to pass the array to the function. span removes the need for the second parameter

#### Solution:

1.

Text

Description automatically generatedText

Description automatically generated

2.

Text

Description automatically generatedText

Description automatically generated

#### Test data:

n/a

#### Sample output:

1.



2.



#### Reflection:

Fairly straight forward there isn’t much to comment on. Span defiantly seems like it would be the way I would want to do it if it’s possible to.

#### Metadata:

Arrays

#### Further information:

n/a

## Week 5 – Lab E

Date: 01/11/2022

### Q1. Basic vectors

#### Question:

1. Add cross product and dot product methods to the given ‘Vector3d’ class.

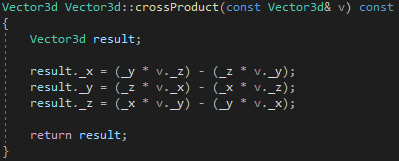
2. Write new methods to overload the basic binary operators.

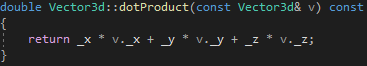
3. Overload the ‘-‘ unary operator so that it inverts the vector.

4. Use timing code to analyse the performance of each implementation of adding 2 vectors.

#### Solution:

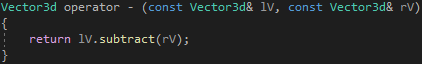
1.





2.

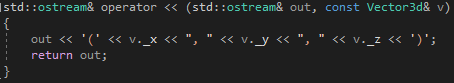
Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generated

3.

Text

Description automatically generated

4.

Graphical user interface, text

Description automatically generated Graphical user interface, text

Description automatically generated Graphical user interface, text

Description automatically generated

#### Test data:



#### Sample output:

1.

 = 

 = 

2.

 = 

 = 

 = 

 = 

 = 

 = 

 = 

Text

Description automatically generated = A picture containing graphical user interface

Description automatically generated

3.

 = 

4.

Release x64

.add()

Text

Description automatically generated

+

Text

Description automatically generated with medium confidence

+=

Text

Description automatically generated

#### Reflection:

This was a pretty easy task to do. In terms of the timings for the addition it is as I would have expected for the most part with operators being faster weirdly enough though += seems to be faster than +.

#### Metadata:

Vector maths

#### Further information:

Some methods are inline to remove the associated overhead which will improve execution time

Why exactly are operators faster than a method? The operators are calling the method anyway

Why is += faster the +?

### Q2. Commutativity

#### Question:

Implement both a standard method and overload the \* operator to multiply a vector by a single double. Also implement the multiplication of a single double by a vector.

#### Solution:

Text

Description automatically generated

Text

Description automatically generated

#### Test data:

Text

Description automatically generated Graphical user interface, text

Description automatically generated

#### Sample output:



#### Reflection:

This was pretty straight forward there isn’t a lot to really comment on

#### Metadata:

Scalar multiply

#### Further information:

n/a

### Q3. Matrices

#### Question:

Add the functionality to do the following:

1. Addition
2. Subtraction
3. Multiplication
4. Streaming in and out
5. Inverse
6. Transpose

#### Solution:

1.

Calendar

Description automatically generatedText

Description automatically generated

2.

Calendar

Description automatically generated with medium confidenceText

Description automatically generated

3.

Calendar

Description automatically generatedText

Description automatically generated

4.

Text

Description automatically generated

5.

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

6.

Calendar

Description automatically generated with medium confidence

#### Test data:

#### Sample output:

 = Text

Description automatically generated

 = Graphical user interface, text

Description automatically generated

 = A picture containing text, outdoor

Description automatically generated

 = A picture containing text, electronics, close

Description automatically generated

 = A picture containing text, electronics, black, keyboard

Description automatically generated

 = Text

Description automatically generated with medium confidence

 = Text

Description automatically generated

 = A picture containing text, electronics, keyboard

Description automatically generated

#### Reflection:

This was a complicated task purely because of some of the maths involved especially inversing. I struggled to find an appropriate operator to overload for transposing. Is it sometimes better to use a method call over an operator? Operators are quicker but in the case of the transpose it can be less clear what is going on.

#### Metadata:

Matrix maths

#### Further information:

Why does  work but  doesn’t? I thought . and -> were the same, is it because is a method call within a class

### Q4. Vector and Matrix Multiplication

#### Question:

Expand your Matrix33d class to be able to multiple a Vector3d object by a Matrix33d object

#### Solution:

Text

Description automatically generatedText

Description automatically generated

#### Test data:

#### Sample output:

Text

Description automatically generated = 

A screenshot of a computer

Description automatically generated with low confidence = 

#### Reflection:

This was very simple assuming I’ve understood the task correctly. It seems too simple so I have a feeling I’ve misunderstood something

#### Metadata:

Vector \* Matrix

#### Further information:

n/a

### Q5. Internal data structures

#### Question:

implement the Matrix33d using a different data format and assess the performance using the timing code from earlier labs

#### Solution:

A screenshot of a computer

Description automatically generated with medium confidenceText

Description automatically generatedCalendar

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidenceCalendar

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedCalendar

Description automatically generatedText

Description automatically generated

#### Test data:

n/a

#### Sample output:

Array of Vector3d

|  |  |
| --- | --- |
| Addition | Text  Description automatically generated with medium confidence |
| Subtraction | A screenshot of a computer  Description automatically generated with medium confidence |
| Multiply Matrices | Text  Description automatically generated with medium confidence |

Multidimensional array of doubles

|  |  |
| --- | --- |
| Addition | Text  Description automatically generated |
| Subtraction | Text  Description automatically generated |
| Multiply Matrices | Text  Description automatically generatedText  Description automatically generated |

#### Reflection:

I decided that the best data structure to implement would be a multidimensional array of doubles, this is essentially the same as the original implementation which was and array of ‘Vector3d’ object. I think the multidimensional array is a lot easier to read and understand though. When looking at timings it seems multidimensional is quicker but the timing was pretty inconsistent so I’m not sure if it is

#### Metadata:

Matrix data structures

#### Further information:

n/a

## Week 6 – Lab F

Date: 08/11/2022

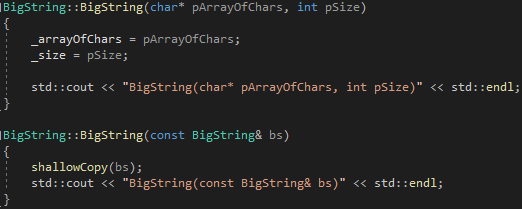
### Q1. Big strings

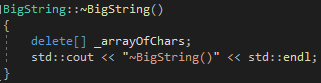
#### Question:

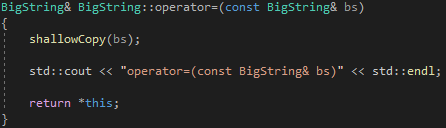
Expand the “BigString” class to contain at least the following functionality:

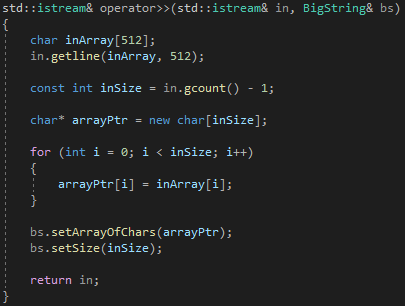
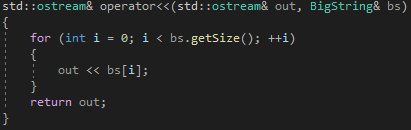
1. Constructors
2. Destructor
3. Assignment operator
4. Stream in and out
5. Index operator

#### Solution:









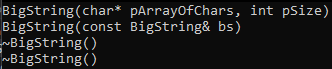


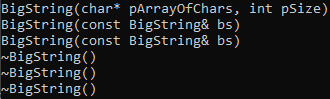
#### Test data:

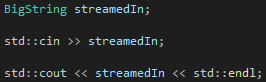


#### Sample output:

 = 

 = 

 = 

 = 

#### Reflection:

I don’t think I really understand destructors that well, I was having a problem with mine but I eventually fixed it. I’m unsure when to use shallow copies or deep copies, deep copies seem superior

#### Metadata:

Gang of three

#### Further information:

When would you want to use shallow copies instead of deep copies?

### Q2. Test harness

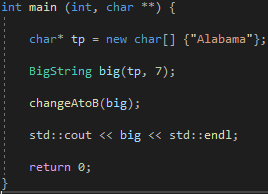
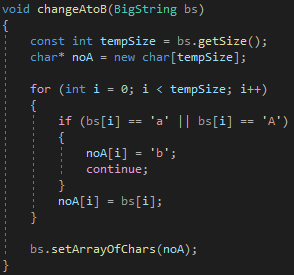
#### Question:

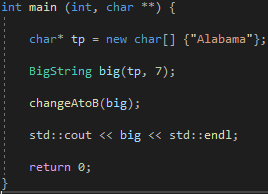
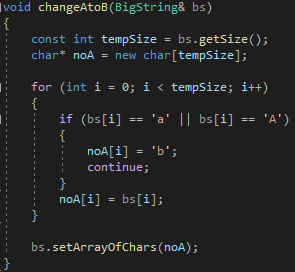
Create code to test all the functionality within BigString. Include at least the following tests:

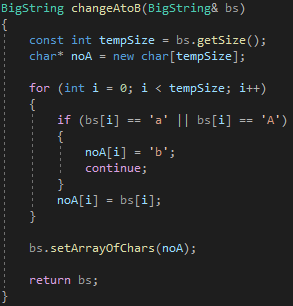
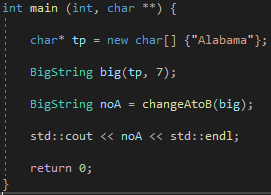
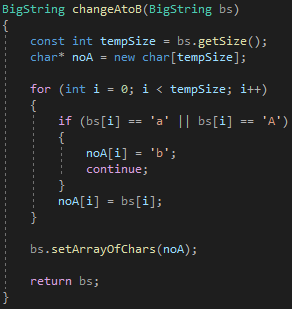
1. Pass BigString to a function, by value
2. Pass BigString to a function, by reference
3. Return BigString from a function, by value
4. Return BigString from a function, by reference
5. Assign one BigString object to another

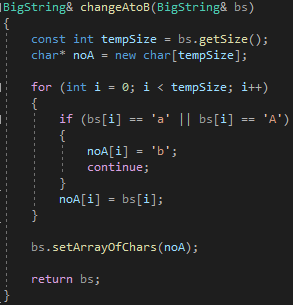
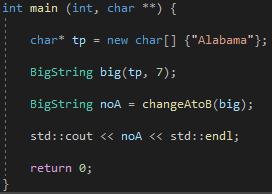
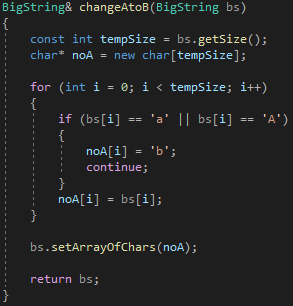
Also check for possible memory leaks

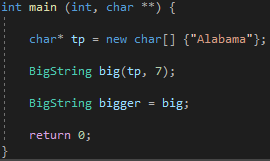
#### Solution:







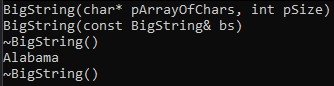




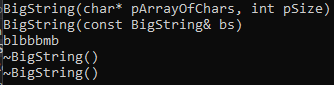
#### Test data:

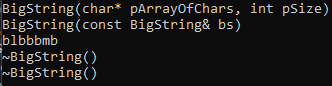
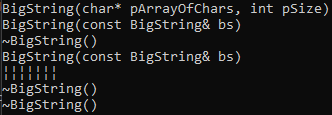


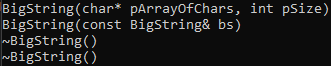
#### Sample output:



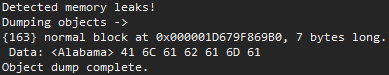








Memory Leak Detecting



#### Reflection:

I’m not sure what part 5 is really expecting, isn’t it just the same as what I have done for the assignment operator testing? Is it to make us think about shallow and deep copies?

I also struggle to understand detecting memory leeks, I think I understand what a memory leek is (solution 1 of part 4 is an example of a memory leek) but the detecting part is confusing to me, **I was doing it right I was just looking in the wrong place**.

#### Metadata:

Testing

#### Further information:

Can you go through memory leek testing briefly?

### Q3. Optimisation

#### Question:

Now that you have a clear picture of which BigString functions are being called; are there any situations where you think you can improve the performance of your code?

#### Solution:

n/a

#### Test data:

n/a

#### Sample output:

n/a

#### Reflection:

The only optimisation I can really think of is maybe using uniform initialization for the default constructor, I’m not sure if this actually has an effect on performance though. We could potentially overload the new and delete operators as well but I don’t understand this enough to be able to say if it would improve performance

#### Metadata:

Optimise

#### Further information:

Are there ways to optimise

## Week 7 – Lab G

Date: 15/11/2022

### Q1. Benchmarks

#### Question:

Produce a set of reliable, reproducible and effective benchmarks.

#### Solution:

Vector3d Benchmark



Matrix33d Benchmark



#### Test data:

Text

Description automatically generated

Calendar

Description automatically generated

#### Sample output:

Vector3d Benchmark Results

Text

Description automatically generated

Matrix33d Benchmark Results

Text

Description automatically generated with low confidence

#### Reflection:

There isn’t much to comment on with this task. The Vector3d class took a bit more to get the timing up to a suitable level for a benchmark I assume that’s just because the matrix maths is more expensive.

#### Metadata:

Benchmarks

#### Further information:

n/a

### Q2. SIMD

#### Question:

Convert your vector and matrix classes to use the SIMD instructions. Time your results. How did they compare to your original implementation?

#### Solution:

Vector SIMD

Text

Description automatically generatedText

Description automatically generated

#### Test data:

Previously outlined benchmarks

#### Sample output:

Vector SIMD Timing

Text

Description automatically generated

Matrix SIMD Timing

Text

Description automatically generated

#### Reflection:

I must be doing something wrong because instead of getting quicker each got 4-6 times slower. Maybe I’m not targeting the right CPU architecture? Although I did double check that I was.

#### Metadata:

SIMD Intrinsic

#### Further information:

What am I doing wrong? Is it that I’m not targeting the correct architecture or something? **It is expected that you implementation will be slower, the loading in your solution is bad as it relies on x, y, z to be next to each other in memory.**

### Q3. Profilers

#### Question:

Profile the Raytracer, using both a sampling and instrumented profiler.

#### Solution:

Sampling

Text

Description automatically generated

Instrumented

Graphical user interface, text, application, email

Description automatically generated

#### Test data:

n/a

#### Sample output:

Sampling

A screenshot of a computer

Description automatically generated with medium confidence

Instrumented

Graphical user interface, text

Description automatically generated

#### Reflection:

This was easy to do once I got VTune installed and found the performance profiler, it was in a different place than described in the video **Debug -> Performance Profiler.** I think I prefer the Microsoft profiler better, something about it is easier to read for me. Although intel definitely gives more information.

#### Metadata:

Profiling

#### Further information:

Which is better instrumentation or sampling?  **It depends on what you are after instrumentation is more detailed**

## Week 8 – Lab H

Date: 22/11/2022

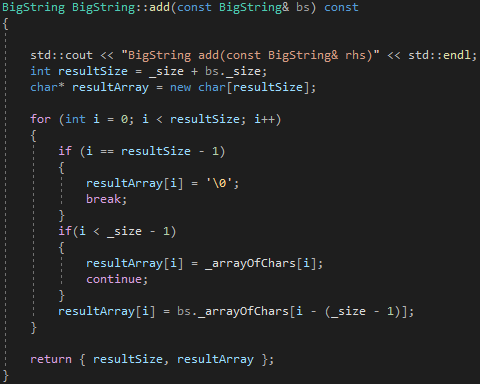
### Q1. BigString concatenators

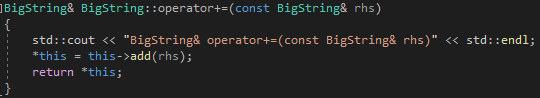
#### Question:

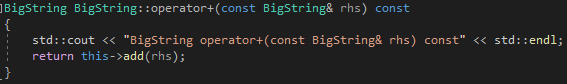
Extend your BigString class to include the following new methods:

1. BigString& operator+= (const BigString& rhs);
2. BigString operator+ (const BigString& rhs) const;

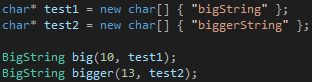
#### Solution:



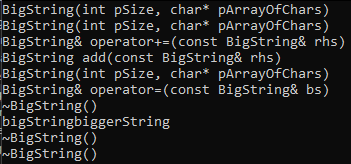


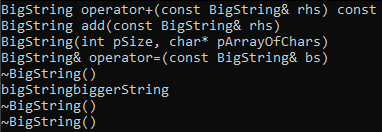


#### Test data:



#### Sample output:

 = 

 = 

#### Reflection:

I couldn’t get the + operator to use += like I wanted while also sticking to the specified prototypes. I’m not sure if I am just not understanding something full or if its not possible will the given prototype.

#### Metadata:

Adding strings

#### Further information:

Am I not understanding something with the + operator?

### Q2. Test harness (BigString concatenators)

#### Question:

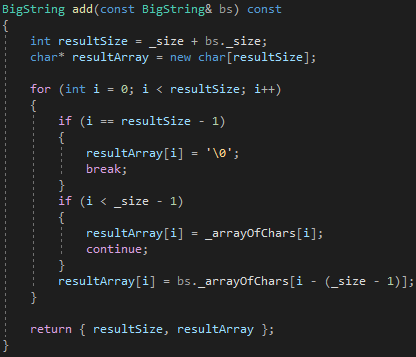
Extend your test hardness to include the new string concatenators, and test the two new methods. Consider ways to improve efficiency then implement and test the improved version.

#### Solution:

Addition benchmark



Inline solution



#### Test data:

n/a

#### Sample output:

Addition benchmark







**Average 3333.33 nanoseconds**

Inline solution







**Average 2900 nanoseconds**

#### Reflection:

The only possible improvement I could think of was to inline the add method. This caused a slight increase in speed but this was probably just lucky timings.

#### Metadata:

Improvement?

#### Further information:

What is the improvement that was expected here?

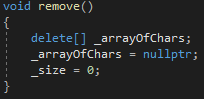
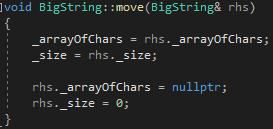
### Q3. BigString move constructor and operator

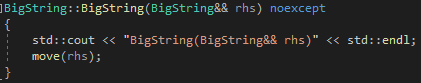
#### Question:

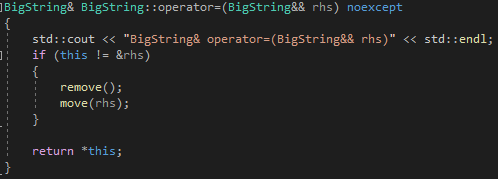
Extend your BigString class to include the following new methods:

1. BigString (const BigString&& rhs);
2. BigString& operator= (const BigString&& rhs);

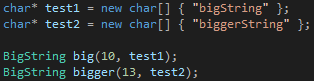
#### Solution:



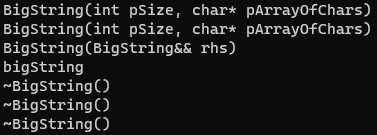


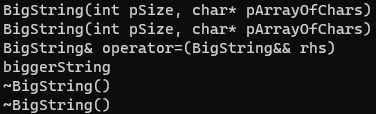


#### Test data:



#### Sample output:

 = 

 = 

#### Reflection:

I think I understand moves I’m just not sure why or when you would need to use them.

#### Metadata:

Moves

#### Further information:

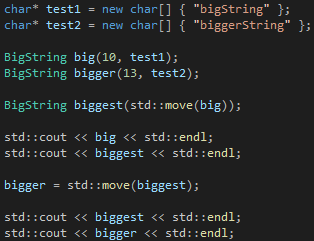
What are some use cases for moves?

### Q4. Test harness (BigString move constructor and operator)

#### Question:

Extend your test hardness to include the two new methods. Evaluate how much performance improvement you get using the move rather than classical functions.

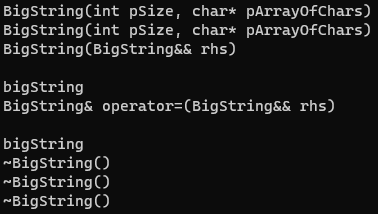
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

I had a misunderstanding of how to use move properly but now I understand that its should be used implicitly by the compiler when there is a temporary anonymous

#### Metadata:

Moves

#### Further information:

n/a

# Final Lab

## Graphics (700106)

* Document and critique the algorithms used, including:
* Geometry representation and processing
* Shading and lighting
* Shadow generation
* Particle system
* Explain how application objects and their graphics representation are connected; discuss the advantages/disadvantages of your approach
* Explain how application-object behavior is updated and how these updates are propagated to the graphics representation
* Discuss potential extensions (especially with respect to non-implemented items from the advanced feature list) as well as potential scalability issues
* What feature (not necessarily mentioned before) would you have liked to add?

## Design (700120)

* Class diagram(s) containing main classes
* Class diagram(s) containing service/utility classes
* A textual description giving the name, role and responsibilities of each class
* Interaction diagram(s) for significant components of the software design
* A critique of the design, including details on:
* The merits of the design?
* Weaknesses of the design?
* What has changed in the design?
* What would you do in a different way?