**Tyler Paulley**

**This test is designed to give us a basic understanding of your programming and design abilities. Answers are expected to be in C/C++. Feel free to insert your answers below the corresponding question or in a separate txt file. Make sure you number your answers if you place them in a separate file. Feel free to add comments to help us understand your thought process for solving these problems.**

**1). Design a bicycle class. Keep the declaration high level.**

\*\*\*\*\* See files in the Exercise1\_Bicycle folder \*\*\*\*\*

Note: These files include a header file and .cpp file for the Bicycle class, and a main.cpp file that tests the class’s functions.

**2). Player A is at Point0(10, 0, 25) in the world and his forward vector is Vector0(2,1,10), Player B is at Point1(-6, 0, 15). Describe how you would determine if Player B is in front of Player A.**

We can determine if PlayerA is facing PlayerB by finding the vector equation formed by Point0 and Vector0.

The vector equation is:

\vec{r} = \vec{r_0} + t \vec{v} 

Where is the vector from the origin (0, 0, 0) to Point0

And is the forward vector, Vector0

Using this, = <10, 0, 25> + t \* <2, 1, 10>

Which gives us the parametric equations:

x = 10 + 2t

y = t

z = 25 + 10t

By using these equations, we can see that PlayerA is not facing PlayerB since the values of Point1 do not solve the equations.

-6 = 10 + 2t

0 = t

15 = 25 + 10t

Using the equation 0 = t, we know that t must = 0 if PlayerA is facing PlayerB. Plugging in t = 0 for the equations yields:

-6 = 10

0 = 0

15 = 25

Since not all of the left and right sides of the equations are equal, we know that PlayerB is not located in the direction that PlayerA is facing.

For example, if PlayerB was located at (30, 2, 55), he would be located in front of PlayerA as seen by:

16 = 10 + 2t

3 = t

55 = 25 + 10t

Which when t = 3 resolves to

16 = 16

3 = 3

55 = 55

Therefore, PlayerB would be in front of PlayerA in this scenario. If the equations solve when t is a negative number, then PlayerB would be behind PlayerA instead of in front of them.

**3). Write an algorithm that reverses the words in a sentence. Example “This world is but a canvas to our imagination” and the result should be “imagination our to canvas a but is world This”.**

**void WordReverse(char \* szWords, const int nSize);**

\*\*\*\*\* See the ReverseWords.cpp file in the Exercise3\_ReverseWords folder \*\*\*\*\*

**4). Correct the logic errors in the code below:**

**char szMessage[] = "Who wants Toast!!!";**

**char \* begin = szMessage;**

**char \* end = szMessage + sizeof(szMessage) / sizeof(szMessage[0]) -2;**

**while(begin < end)**

**{**

**char swap = \*begin;**

**\*begin++ = \*end;**

**\*end++ = swap;**

**}**

**Result should be: “!!!tsaoT stnaw ohW”**

\*\*\*\*\* See the ReverseString.cpp file in the Exercise4\_CorrectTheCode folder \*\*\*\*\*

**5). Write a function that tests if a point is within a specified distance of any part of a filled rectangle. The rectangle is specified by its center point, extents and rotation.**

**struct s\_Vector**

**{**

**float x;**

**float y;**

**};**

**struct s\_Rectangle**

**{**

**s\_Vector center; // center of the rect in world space**

**s\_Vector localX; // local space X direction vector, normalized**

**s\_Vector localY; // local space Y direction vector, normalized**

**float fExtentsX; // distance from the rect center to the right edge**

**float fExtentsY; // distance from the rect center to the top edge**

**};**

**bool IsPointWithinDistOfRectangle(s\_Rectangle & rect, s\_Vector & point, float distance);**

\*\*\*\*\* See the Rectangle.cpp file in the Exercise5\_Rectangle folder \*\*\*\*\*