**Take Home Questions**

## Question 1

Write a C# program to solve the following problems:

1. Write a function that receives three integer inputs for the lengths of the sides of a triangle and returns the triangle type (scalene, isosceles, equilateral).
2. Write a function that would return the 5th element from the end in a singly linked list of integers, in one pass, and then provide a set of test cases against that function. Assume the list size cannot be known without traversing the list.

## Question 2

Write a C# program to solve the following problems:

1. Take as input the path to a file containing one integer per line. For each integer in the file, output to the console a comma-delimited list of the integer's prime factors. The list of integers on each line of the output should multiply to produce the input integer. Please include unit tests for the code  ([http://www.nunit.org](http://www.nunit.org/)  preferred)

If you think of any other questions, make an assumption and document it in your code

Customers  
ID NAME ADDRESS PHONE NUMBER EMAIL  
  
Orders  
ID CUSTOMER\_ID ORDER\_AMOUNT ORDER\_ADDRESS

1. Write the SQL Query to pull all customers

**Select \* from Customers**

1. Write a SQL Query to pull all customers that have orders (no duplicates)

**Select distinct c.\***

**from Customers c join Orders o on c.ID = o.CUSTOMER\_ID**

1. Write a SQL Query to pull all customers that do NOT have orders

**Select \***

**from Customers**

**where ID not in (select CUSTOMER\_ID from Orders)**

1. If you had to create an index on these tables, which fields would you most likely index and why?

**Name, Email, and Phone\_Number on Customers table. These are high likely searchable. Customer\_ID on Orders table should have a foreign key constraint.**

1. Write a query that lists each customer name, email, and the number of order they have

**Select c.NAME, c.EMAIL, c.PHONE\_NUMBER, Count(o.ID)**

**from Customers c join Orders o**

**group by c.NAME, c.EMAIL, c.PHONE\_NUMBER**

1. Write query that pulls all customers with between 2 and 5 orders.

**Select c.ID, c.NAME, Count(o.ID)**

**from Customers c join Order o**

**group by c.ID, c.NAME**

**having count(o.ID) >1 and count(o.ID) < 6**