**Answers to Questions for Assignment 2, Semester 2, C# Course**

May 24th 2020

**-By Valentin Vasilev - 3860140**

**Question 1.a:**

**Definition of Combinatorial Logic:**

Combinatorial logic is a concept in which two or more input states define one or more output states, where the resulting state or states are related by defined rules that are independent of previous states. Each of the inputs and output(s) can attain either of two states: logic 0 (low) or logic 1 (high).

By: <https://whatis.techtarget.com/definition/combinatorial-logic>

**Question 1.b:**

**My own understanding of it:**

This is a transition that takes 2 or more inputs and outputs their result. It only cares what the inputs are, without their history. This is used for an endless amount of things in the electronics world. All the digital logic is based on Combinatorial Logic. This is what we use for Boolean Algebra.

Things that can be computed with Combinatorial Logic:

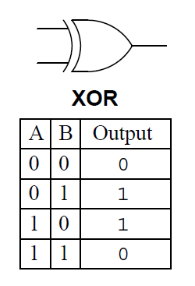
Checking if a few conditions are met or not.

MOSFET Transistors

Things that cannot be computed with combinatorial Logic:

“If it rains, It will probably be cold.”

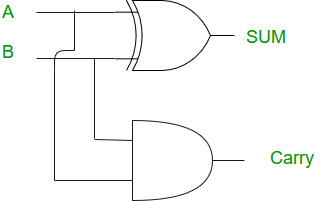
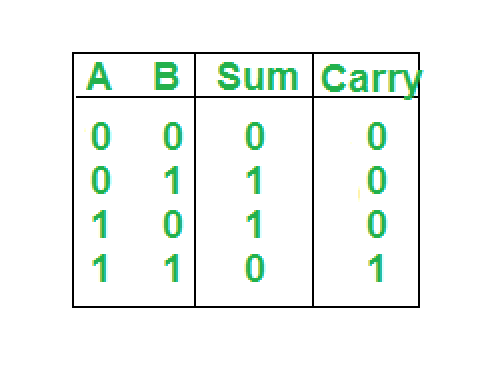
Anything that incorporates randomness.

I’ve actually had a hard time figuring out things that cannot be computed with Combinatorial Logic. 

**Question 3:**

The addition of 2bits is called Half adder the input variables are augent and addent bits and output variebles are sum&carry bits. A and B are the two input bits.

As far as I understand, with a Half Adder we work with and 2 and only 2 inputs. This sounds like a limit, but given we can always add more Half Adders it is unfair to call it a “limit”. It produces 2 outputs – sum and carry bit. This adder is used for a numerous of day-to-day applications. To name a few: Encoders, Decoders.



**Question 4.a:**

**Composition** is one of the fundamental concepts in **object-oriented programming**. It describes a class that references one or more objects of other classes in instance variables. This allows you to model a has-a association between objects.

**Question 4.b & c & d & e:**

I, personally, understand composition in the context of oop as follows:

* I need to be able to use the classed I have already created to simplify my future work
* Classes need to go logically more and more in-depth like a “tree”
* Create things in a way that are most useable by as many other things as possible

For example using my Half Adder code in my Full Adder class to simplify my work process. This is a classic case of composition. I explain it to myself as “If I wrote and did the math already once, why should I do it again if I can use the old solution”.

Advantages are that you simplify your work and decrease your work load by re-using things you have already done.

Disadvantages are probably when you are new to something you do, it takes you more time to remember the more basic things like some basic functions syntax and so on (E.g. learning a new language).

Based on the aforementioned advantage and disadvantages, I have used composition to build the Full Adder, because it is easier for me.