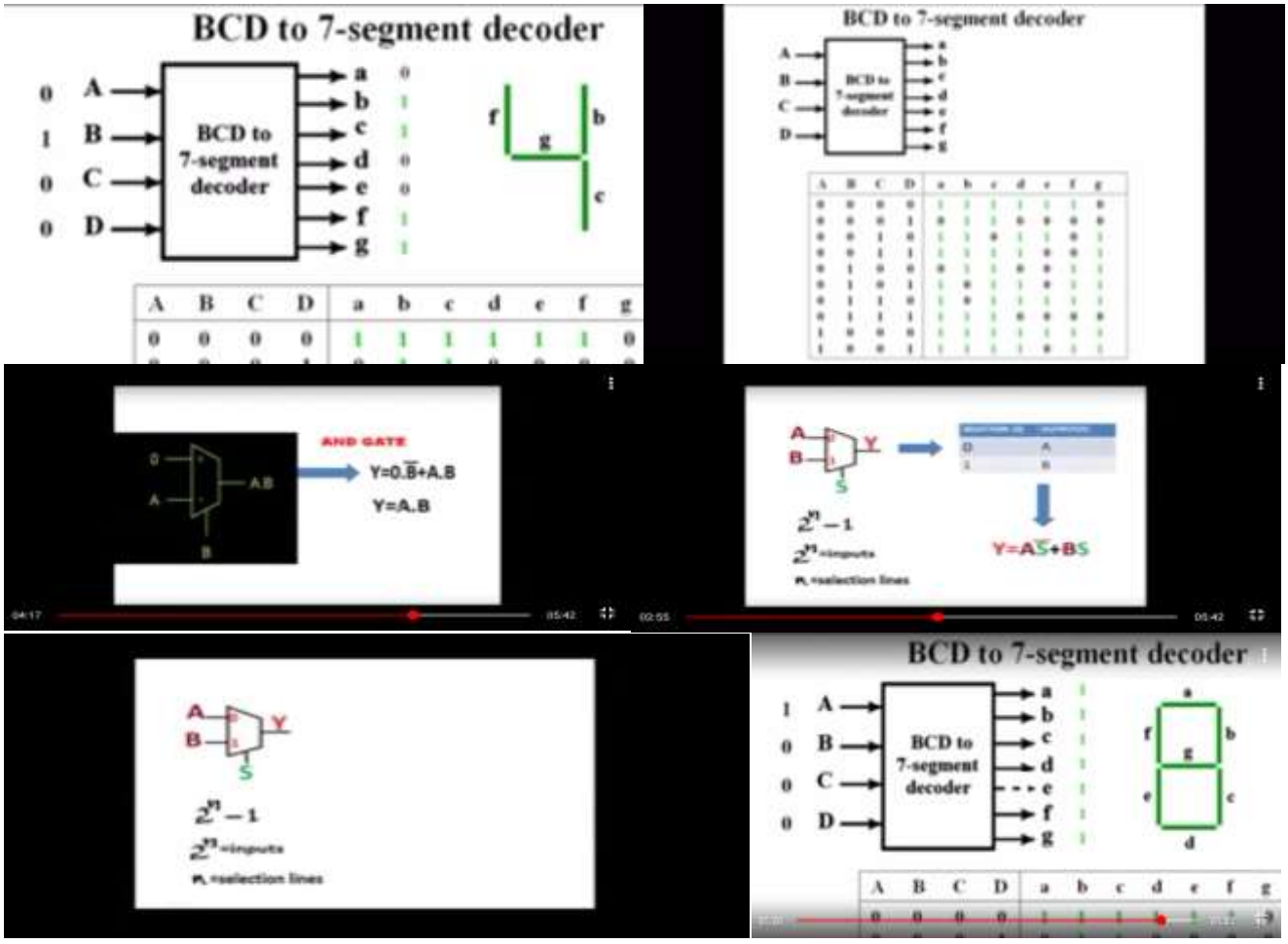
DAILY ASSESSMENT FORMAT

Date:	**28_05_2020**	**Name:**	**PRINCIA MELITA DSOUZA**
Course:	**LOGIC DESIGN**	**USN:**	**4AL17EC075**
Topic:	**BOOLEAN EQUATIONS,COMBINATIONAL CIRCUITS & 7 SEGMENT DECODER**	**Semester & Section:**	**6TH B**
Github Repository:	**MELITA-1999**		

FORENOON SESSION DETAILS

Image of session



Report – Report can be typed or hand written for up to two pages.

DESIGN OF 7 SEGMENT DECODER WITH COMMON ANODE DISPLAY

BCD to 7-Segment Display Decoders
A binary coded decimal (BCD) to 7-segment display decoder such as the TTL 74LS47 or

74LS48, have 4 BCD inputs and 7 output lines, one for each LED segment. This allows a smaller 4-bit binary number (half a byte) to be used to display all the denary numbers from 0 to 9 and by adding two displays together, a full range of numbers from 00 to 99 can be displayed with just a single byte of eight data bits.

BCD to 7-Segment Decoder

The use of packed BCD allows two BCD digits to be stored within a single byte (8-bits) of data, allowing a single data byte to hold a BCD number in the range of 00 to 99.
An example of the 4-bit BCD input (0100) representing the number "4" is given below.

Display Decoder Example No1

In practice current limiting resistors of about 150Ω to 220Ω would be connected in series between the decoder/driver chip and each LED display segment to limit the maximum current flow. There are different display decoders and drivers available for the different types of available displays, either LED or LCD. For example, the 74LS48 for common-cathode LED types, the 74LS47 for common-anode LED types, or the CMOS CD4543 for liquid crystal display (LCD) types.Liquid crystal displays (LCD´s) have one major advantage over similar LED types in that they consume much less power and nowadays, both LCD and LED displays are combined together to form larger Dot-Matrix Alphanumeric type displays which can show letters and characters as well as numbers in standard Red or Tri-colour outputs.

7-Segment Display Elements for all Numbers.

It can be seen that to display any single digit number from 0 to 9 in binary or letters from A hexadecimal, we would require seven separate segment connections plus one additional co the LED's "common" connection. Also as the segments are basically a standard light emit up to 20mA of current to illuminate each individual segment and to display the number "8" lit resulting a total current of nearly 140mA, (8 x 20mA).

Obviously, the use of so many connections and power consumption is impractical for some microprocessor based circuits and so in order to reduce the number of signal lines required one single display, display decoders such as the BCD to 7-Segment Display Decoder and I are used instead.

Binary Coded Decimal

Binary Coded Decimal (BCD or "8421" BCD) numbers are made up using just 4 data bits (half a byte) similar to the Hexadecimal numbers we saw in the binary tutorial, but unlike he numbers that range in full from 0 through to F, BCD numbers only range from 0 to 9, with

number patterns of 1010 through to 1111 (A to F) being invalid inputs for this type of display and s
are not used as shown below.

Display Decoder

A Display Decoder is a combinational circuit which decodes and n-bit input value into a nu
A Digital Decoder IC, is a device which converts one digital format into another and one o
called the Binary Coded Decimal (BCD) to 7-Segment Display Decoder.7-segment LED (L
type displays, provide a very convenient way of displaying information or digital data in th
characters.

Typically 7-segment displays consist of seven individual coloured LED's (called the segme
produce the required numbers or HEX characters from 0 to 9 and A to F respectively, on th
need to be illuminated and BCD to 7-segment Display Decoders such as the 74LS47 do jus
eight (8) input connections, one for each LED segment and one that acts as a common term
Some single displays have also have an additional input pin to display a decimal point in th
In electronics there are two important types of 7-segment LED digital display.

1. The Common Cathode Display (CCD) – In the common cathode display, all the cathode
"0" or ground. The individual segments are illuminated by application of a "HIGH", logic "
2. The Common Anode Display (CAD) – In the common anode display, all the anode conn
and the individual segments are illuminated by connecting the individual Cathode terminals

Combination circuit :

A combination circuit is one that has a "combination" of series and parallel paths for the electricity to flow. Its properties are a combination of the two. In this example, the parallel section of the circuit is like a sub-circuit and actually is part of an over-all series circuit.

BOOLEAN EQUATIONS FOR DIGITAL CIRCUITS

Boolean Algebra is an algebra, which deals with binary numbers & binary variables. Hence
A mathematician, named George Boole had developed this algebra in 1854. The variables u
variables.

The range of voltages corresponding to Logic ‘High’ is represented with ‘1’ and the range of voltag
with ‘0’.

Postulates and Basic Laws of Boolean Algebra
In this section, let us discuss about the Boolean postulates and basic laws that are used in Boolean a
Boolean functions.

Boolean Postulates
Consider the binary numbers 0 and 1, Boolean variable xx and its complement x′x′. Either the Boole
literal. The four possible logical OR operations among these literals and binary numbers are shown

x + 0 = x

x + 1 = 1

x + x = x

x + x’ = 1

Similarly, the four possible logical AND operations among those literals and binary numbers are sh

x.1 = x

x.0 = 0

x.x = x

x.x’ = 0

These are the simple Boolean postulates. We can verify these postulates easily, by substituting the B

Note− The complement of complement of any Boolean variable is equal to the variable itself. i.e., x

Basic Laws of Boolean Algebra
Following are the three basic laws of Boolean Algebra.

Commutative law
Associative law
Distributive law
Commutative Law
If any logical operation of two Boolean variables give the same result irrespective of the order of th

is said to be Commutative. The logical OR & logical AND operations of two Boolean variables x &

$x + y = y + x$

$x.y = y.x$

The symbol '+' indicates logical OR operation. Similarly, the symbol '.' indicates logical AND ope
Commutative law obeys for logical OR & logical AND operations.

Associative Law
If a logical operation of any two Boolean variables is performed first and then the same operation is
the same result, then that logical operation is said to be Associative. The logical OR & logical AND
z are shown below.

$x + y+zy+z = x+yx+y + z$

$x.y.zy.z = x.yx.y.z$

Associative law obeys for logical OR & logical AND operations.

Distributive Law
If any logical operation can be distributed to all the terms present in the Boolean function, then that
The distribution of logical OR & logical AND operations of three Boolean variables x, y & z are sh

$x.y+zy+z = x.y + x.z$

$x + y.zy.z = x+yx+y.x+zx+z$
Distributive law obeys for logical OR and logical AND operations.

These are the Basic laws of Boolean algebra. We can verify these laws easily, by substituting the Bo

Theorems of Boolean Algebra
The following two theorems are used in Boolean algebra.

Duality theorem
DeMorgan's theorem
Duality Theorem
This theorem states that the dual of the Boolean function is obtained by interchanging the logical A
zeros with ones. For every Boolean function, there will be a corresponding Dual function.

Let us make the Boolean equations relationsrelations that we discussed in the section of Boolean po
following table shows these two groups.

Group1 Group2
x + 0 = x x.1 = x
x + 1 = 1 x.0 = 0
x + x = x x.x = x
x + x’ = 1 x.x’ = 0
x + y = y + x x.y = y.x
x + y+zy+z = x+yx+y + z x.y.zy.z = x.yx.y.z
x.y+zy+z = x.y + x.z x + y.zy.z = x+yx+y.x+zx+z
In each row, there are two Boolean equations and they are dual to each other. We can verify all thes
by using duality theorem.

DeMorgan’s Theorem
This theorem is useful in finding the complement of Boolean function. It states that the complement
variables is equal to the logical AND of each complemented variable.

DeMorgan’s theorem with 2 Boolean variables x and y can be represented as

x+yx+y’ = x’.y’

The dual of the above Boolean function is

x.yx.y’ = x’ + y’

Therefore, the complement of logical AND of two Boolean variables is equal to the logical OR of e
apply DeMorgan’s theorem for more than 2 Boolean variables also.

Simplification of Boolean Functions
Till now, we discussed the postulates, basic laws and theorems of Boolean algebra. Now, let us simp

Example 1
Let us simplify the Boolean function, f = p’qr + pq’r + pqr’ + pqr

We can simplify this function in two methods.

Method 1

Given Boolean function, f = p’qr + pq’r + pqr’ +pqr.

Step 1 − In first and second terms r is common and in third and fourth terms pq is common. So, take

⇒ f = p′q+pq′p′q+pq′r + pqr′+rr′+r

Step 2 − The terms present in first parenthesis can be simplified to Ex-OR operation. The terms pres
‘1’ using Boolean postulate

⇒ f = p⊕qp⊕qr + pq11
Step 3 − The first term can’t be simplified further. But, the second term can be simplified to pq usin

⇒ f = p⊕qp⊕qr + pq

Therefore, the simplified Boolean function is f = p⊕qp⊕qr + pq

Method 2

Given Boolean function, f = p’qr + pq’r + pqr’ + pqr.

Step 1 − Use the Boolean postulate, x + x = x. That means, the Logical OR operation with any Bool
same variable. So, we can write the last term pqr two more times.

⇒ f = p’qr + pq’r + pqr’ + pqr + pqr + pqr

Step 2 − Use Distributive law for 1st and 4th terms, 2nd and 5th terms, 3rd and 6th terms.

⇒ f = qrp′+pp′+p + prq′+qq′+q + pqr′+rr′+r
Step 3 − Use Boolean postulate, x + x’ = 1 for simplifying the terms present in each parenthesis.

⇒ f = qr11 + pr11 + pq11
Step 4 − Use Boolean postulate, x.1 = x for simplifying the above three terms.

⇒ f = qr + pr + pq

⇒ f = pq + qr + pr

Therefore, the simplified Boolean function is f = pq + qr + pr.

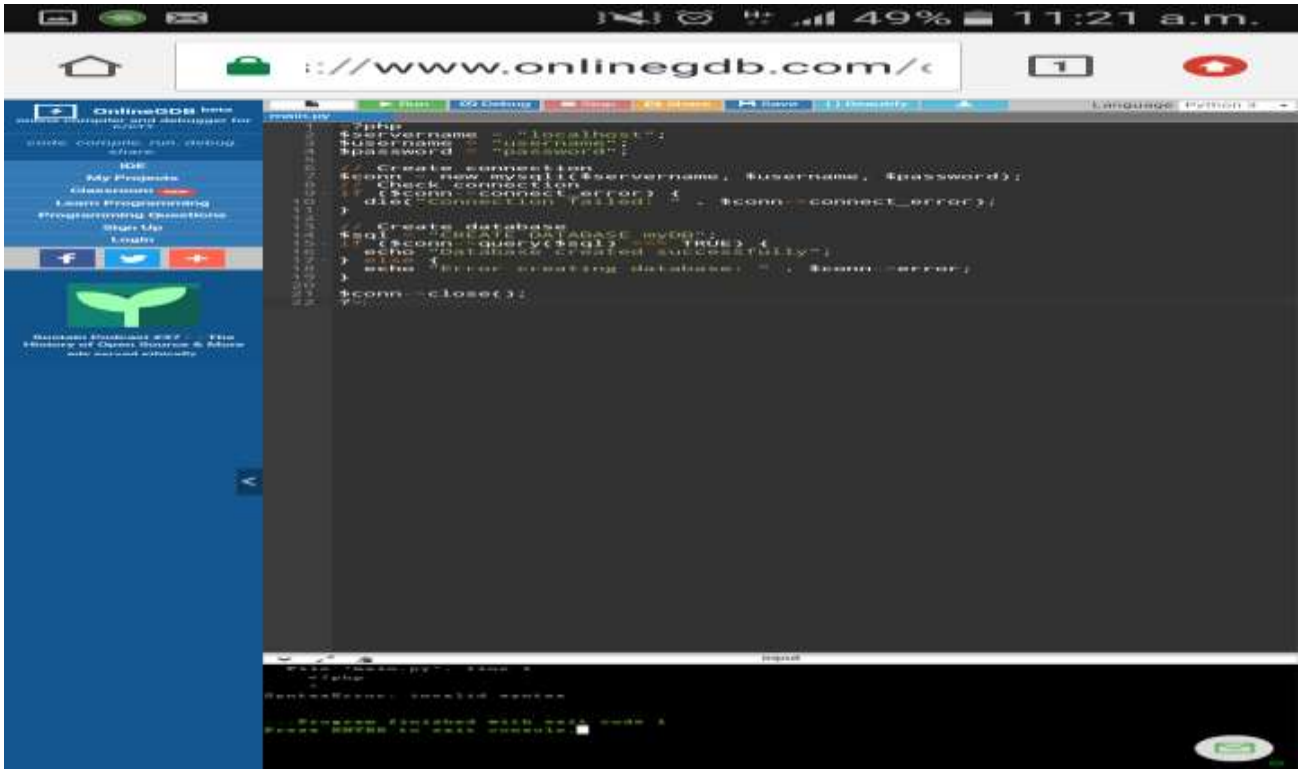
So, we got two different Boolean functions after simplifying the given Boolean function in each me
functions are same. So, based on the requirement, we can choose one of those two Boolean function

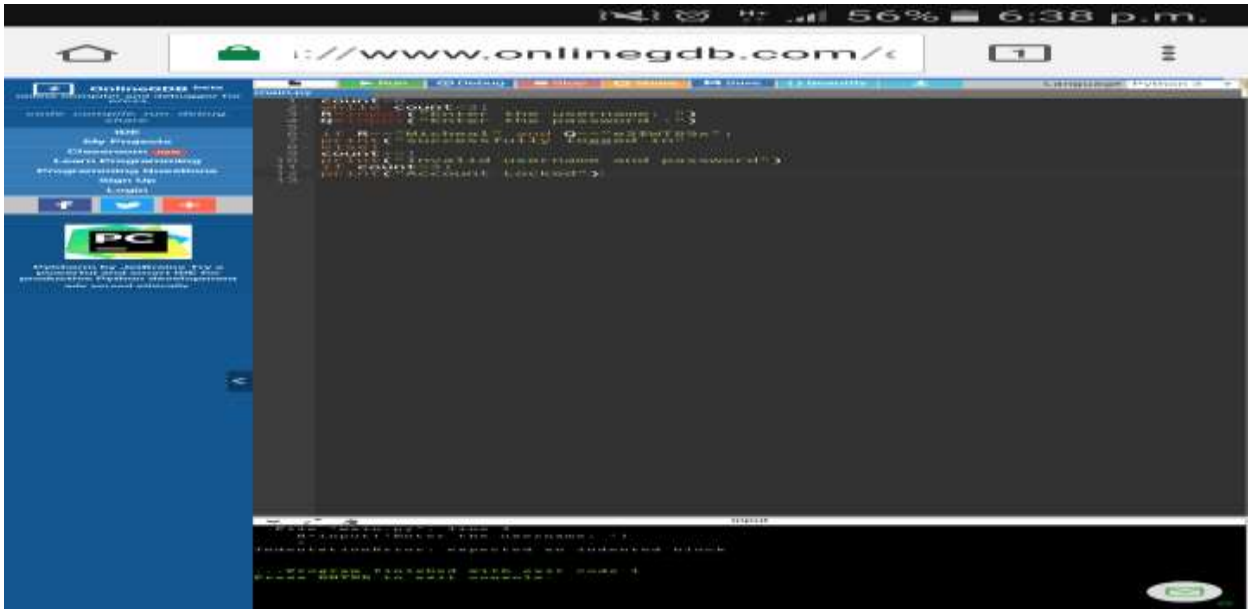
Date:	**28_05_2020**	**Name:**	**PRINCIA MELITA DSOUZA**
Course:	**PYTHON**	**USN:**	**4AL17EC075**
Topic:	**BUILD A DESKTOP DATABASE APPLICATION**	**Semester & Section:**	**6TH B**

Section:

AFTERNOON SESSION DETAILS

Image of session





Report – Report can be typed or hand written for up to two pages.

CREATE DATABASE

A Database is defined as a structured set of data. So, in SQL the very first step to store the data in a well structured manner is to create a database. The CREATE DATABASE statement is used to create a new database in SQL.

Syntax:

CREATE DATABASE database_name;

database_name: name of the database.
Example Query:
This query will create a new database in SQL and name the database as my_database.

CREATE DATABASE my_database;
CREATE TABLE

We have learned above about creating databases. Now to store the data we need a table to do that. The CREATE TABLE statement is used to create a table in SQL. We know that a table comprises of rows and columns. So while creating tables we have to provide all the information to SQL about the names of the columns, type of data to be stored in columns, size of the data etc. Let us now dive into details on how to use CREATE TABLE statement to create tables in SQL.

Syntax:

```
CREATE TABLE table_name
(
column1 data_type(size),
column2 data_type(size),
column3 data_type(size),
....
);
```

table_name: name of the table.
column1 name of the first column.
data_type: Type of data we want to store in the particular column.
For example,int for integer data.
size: Size of the data we can store in a particular column. For example if for a column we specify the data_type as int and size as 10 then this column can store an

integer
number of maximum 10 digits.
OBJ

Example Query:
This query will create a table named Students with three columns, ROLL_NO, NAME and SUBJECT.

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
);
```

This query will create a table named Students. The ROLL_NO field is of type int and can store an integer number of size 3. The next two columns NAME and SUBJECT are of type varchar and can store characters and the size 20 specifies that these two fields can hold maximum of 20 characters.