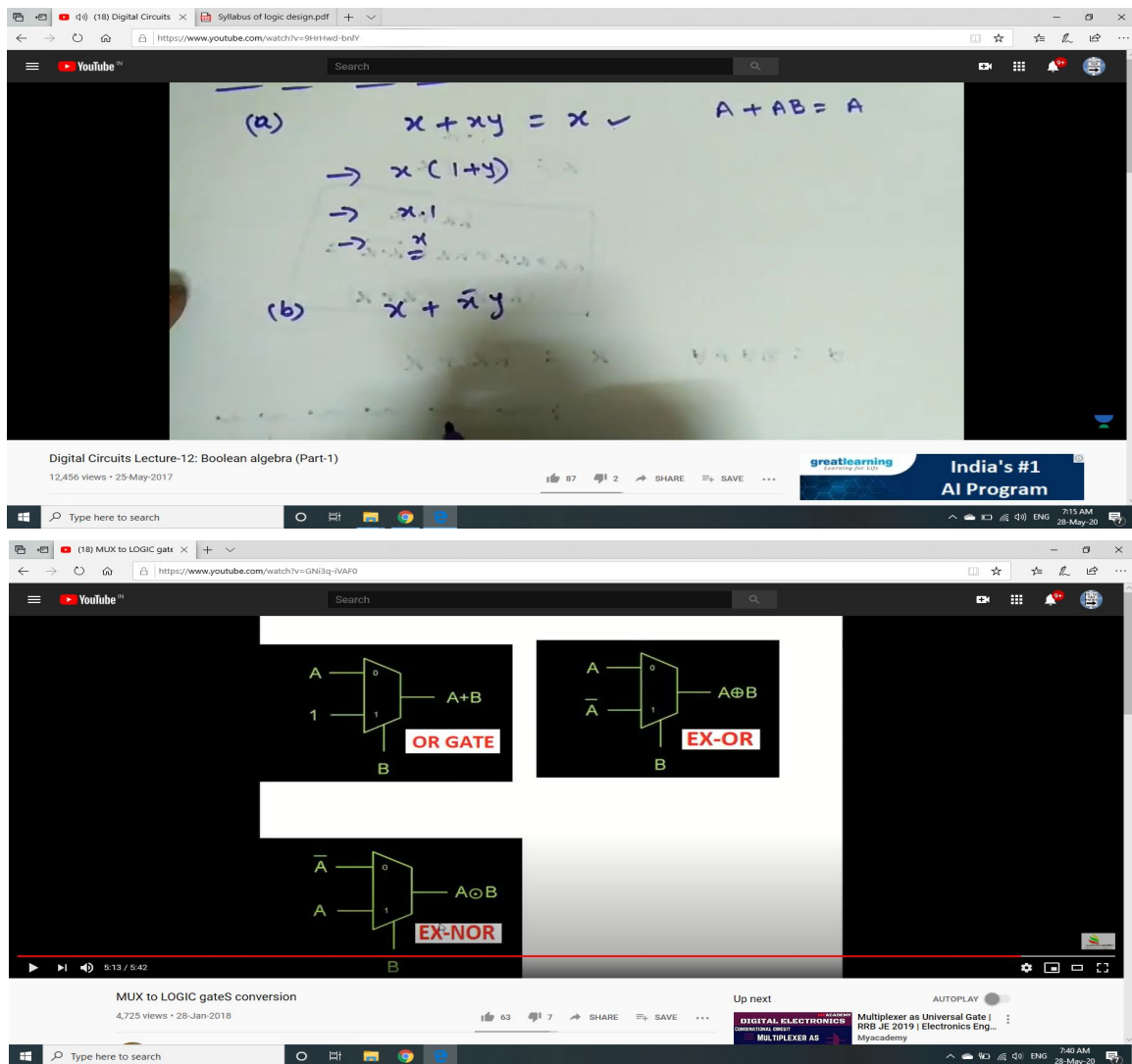


# REPORT MAY 28

Date:	28 MAY 2020	Name:	Rakshith B
Course:	Logic Design	USN:	4AL16EC409
Topic:	Boolean equations for digital circuits. Combinational circuits: Conversion of MUX and Decoders to logic gates.	Semester & Section:	6th SEM B
Github Repository:	Rakshith-B		

## FORENOON SESSION DETAILS

### Image of session



## Report –

### Boolean Algebra :

- In 1854 ,George Boole Developed an Algebraic System Called Boolean Algebra.
- Boolean Algebra is a System of Mathematical Logics.
- It is Defined With a set of Elements,a set of Operators and a Number of Postulates

#### Laws of Boolean Algebra:

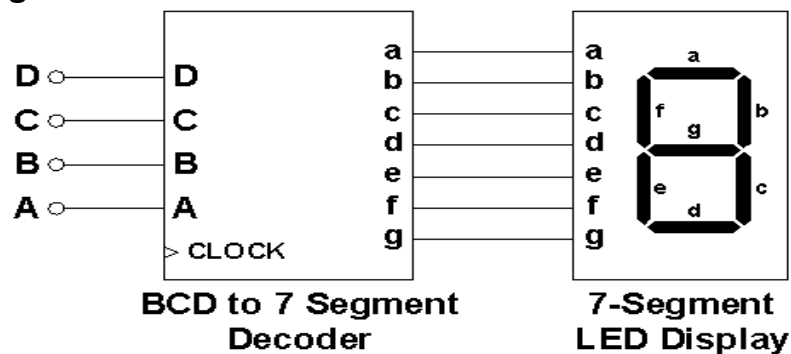
- Commutative Law  
 $X+Y=Y+X$   
 $A+B =B+A$   
 $X.Y=Y.X$   
 $A.B=B.A$
- Associative Law  
 $X+(Y+Z)=(X+Y)+3$   
 $A+(B+C)=(A+B)+C$   
 $X.(Y.Z)=(X.Y).Z$   
 $A.(B.C)=(A.B).C$
- Distributive Law  
 $X(Y+Z)=XY+YZ$   
 $A(B+C)=AB+AC$
- Absorption Theorem  
 $X+XY=X$   
 $A+AB=A$   
 $X+\sim XY=X+Y$

### MUX to Logic Gates Conversion:

- 1.NAND,NOR-Universal Gates
- 2.Universal Gates
- 3.MUX and Decoders are Called Universal Logic

Multiplexer is device which Selects one or Several Digital or Analog Inputs and It will Forward it to the Output Line, Which is Single Output line.

### BCD to SEVEN Segment Decoder



Binary Inputs	Decoder Outputs	7 Segment Display Outputs
D C B A	a b c d e f g	
0 0 0 0	1 1 1 1 1 1 0	0
0 0 0 1	0 1 1 0 0 0 0	1
0 0 1 0	1 1 0 1 1 0 1	2
0 0 1 1	1 1 1 1 0 0 1	3
0 1 0 0	0 1 1 0 0 1 1	4
0 1 0 1	1 0 1 1 0 1 1	5
0 1 1 0	1 0 1 1 1 1 1	6
0 1 1 1	1 1 1 0 0 0 0	7
1 0 0 0	1 1 1 1 1 1 1	8
1 0 0 1	1 1 1 1 0 1 1	9

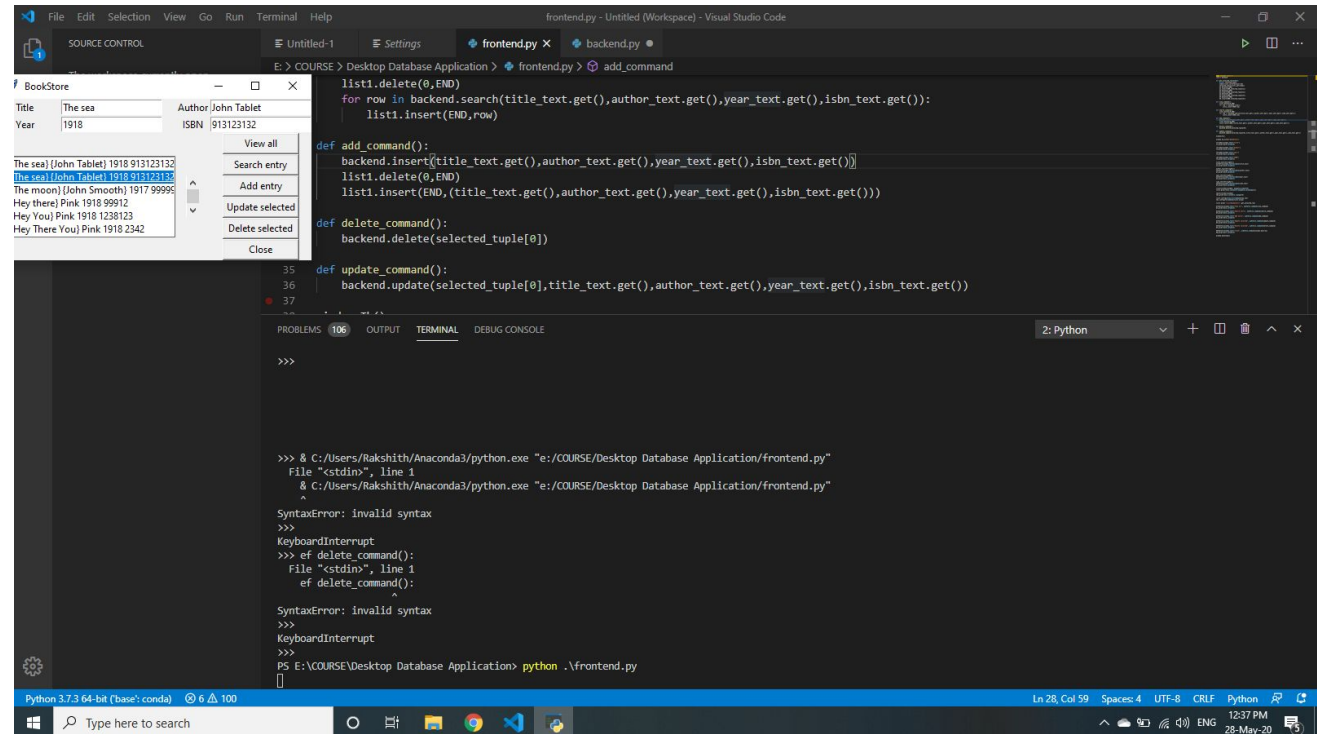
Quickgrid

Date: 28 MAY 2020  
Course: Python On Udemy  
Topic: Build a Desktop Database Application

Name: RAKSHITH B  
USN: 4AL16EC409  
Semester & Section: 6 B

## AFTERNOON SESSION DETAILS

### Image of session: Output



## Report – Frontend.py

```
from tkinter import *
import backend

def get_selected_row(event):
    global selected_tuple
    index=list1.curselection()[0]
    selected_tuple=list1.get(index)
    e1.delete(0,END)
    e1.insert(END,selected_tuple[1])
    e2.delete(0,END)
    e2.insert(END,selected_tuple[2])
    e3.delete(0,END)
    e3.insert(END,selected_tuple[3])
    e4.delete(0,END)
    e4.insert(END,selected_tuple[4])

def view_command():
    list1.delete(0,END)
    for row in backend.view():
        list1.insert(END,row)

def search_command():
    list1.delete(0,END)
    for row in
backend.search(title_text.get(),author_text.get(),year_text.get(),isbn_text.g
et()):
        list1.insert(END,row)

def add_command():
    backend.insert(title_text.get(),author_text.get(),year_text.get(),isbn_text.g
et())
    list1.delete(0,END)

list1.insert(END,(title_text.get(),author_text.get(),year_text.get(),isbn_text.get()))
```

```
def delete_command():
    backend.delete(selected_tuple[0])

def update_command():
    backend.update(selected_tuple[0], title_text.get(), author_text.get(), year_text.get(), isbn_text.get())

window=Tk()

window.wm_title("BookStore")

l1=Label(window, text="Title")
l1.grid(row=0, column=0)

l2=Label(window, text="Author")
l2.grid(row=0, column=2)

l3=Label(window, text="Year")
l3.grid(row=1, column=0)

l4=Label(window, text="ISBN")
l4.grid(row=1, column=2)

title_text=StringVar()
e1=Entry(window, textvariable=title_text)
e1.grid(row=0, column=1)

author_text=StringVar()
e2=Entry(window, textvariable=author_text)
e2.grid(row=0, column=3)

year_text=StringVar()
e3=Entry(window, textvariable=year_text)
e3.grid(row=1, column=1)

isbn_text=StringVar()
e4=Entry(window, textvariable=isbn_text)
e4.grid(row=1, column=3)
```

```

list1=Listbox(window, height=6,width=35)
list1.grid(row=2,column=0,rowspan=6,columnspan=2)

sb1=Scrollbar(window)
sb1.grid(row=2,column=2,rowspan=6)

list1.configure(yscrollcommand=sb1.set)
sb1.configure(command=list1.yview)

list1.bind('<<ListboxSelect>>',get_selected_row)

b1=Button(window,text="View all", width=12,command=view_command)
b1.grid(row=2,column=3)

b2=Button(window,text="Search entry", width=12,command=search_command)
b2.grid(row=3,column=3)

b3=Button(window,text="Add entry", width=12,command=add_command)
b3.grid(row=4,column=3)

b4=Button(window,text="Update selected", width=12,command=update_command)
b4.grid(row=5,column=3)

b5=Button(window,text="Delete selected", width=12,command=delete_command)
b5.grid(row=6,column=3)

b6=Button(window,text="Close", width=12,command=window.destroy)
b6.grid(row=7,column=3)

window.mainloop()

```

#### Backend:

```

import sqlite3

def connect():
    conn=sqlite3.connect("books.db")
    cur=conn.cursor()

```

```

        cur.execute("CREATE TABLE IF NOT EXISTS book (id INTEGER PRIMARY KEY,
title text, author text, year integer, isbn integer)")
        conn.commit()
        conn.close()

def insert(title,author,year,isbn):
    conn=sqlite3.connect("books.db")
    cur=conn.cursor()
    cur.execute("INSERT INTO book VALUES
(NULL,?,?,?,?,?)", (title,author,year,isbn))
    conn.commit()
    conn.close()
    view()

def view():
    conn=sqlite3.connect("books.db")
    cur=conn.cursor()
    cur.execute("SELECT * FROM book")
    rows=cur.fetchall()
    conn.close()
    return rows

def search(title="",author="",year="",isbn=""):
    conn=sqlite3.connect("books.db")
    cur=conn.cursor()
    cur.execute("SELECT * FROM book WHERE title=? OR author=? OR year=? OR
isbn=?", (title,author,year,isbn))
    rows=cur.fetchall()
    conn.close()
    return rows

def delete(id):
    conn=sqlite3.connect("books.db")
    cur=conn.cursor()
    cur.execute("DELETE FROM book WHERE id=?", (id,))
    conn.commit()
    conn.close()

def update(id,title,author,year,isbn):

```



```
conn=sqlite3.connect("books.db")
cur=conn.cursor()
cur.execute("UPDATE book SET title=?, author=?, year=?, isbn=? WHERE
id=?", (title,author,year,isbn,id))
conn.commit()
conn.close()

connect()
insert("The Sun","John Smith",1918,913123132)
delete(3)
update(4,"The moon","John Smooth",1917,999999)
print(view())
print(search(author="John Smooth"))
```