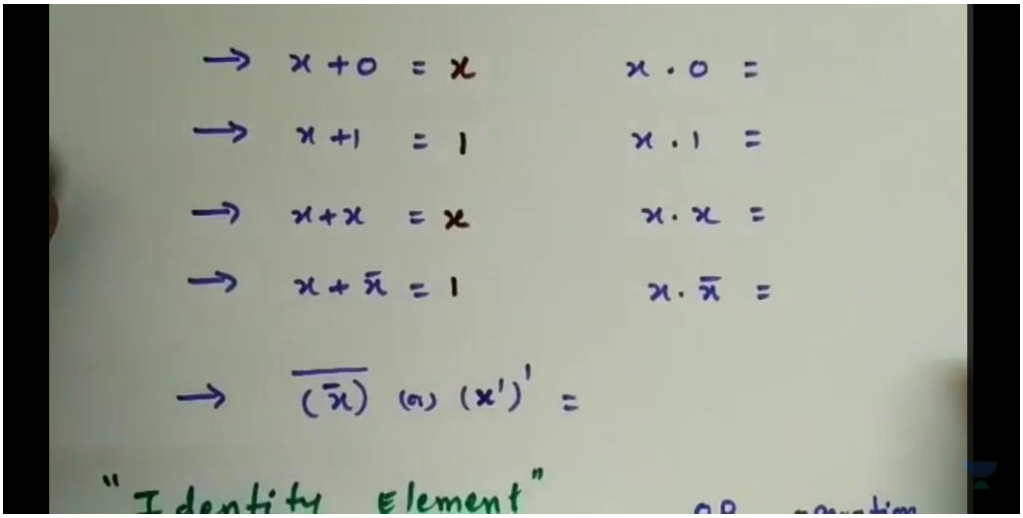
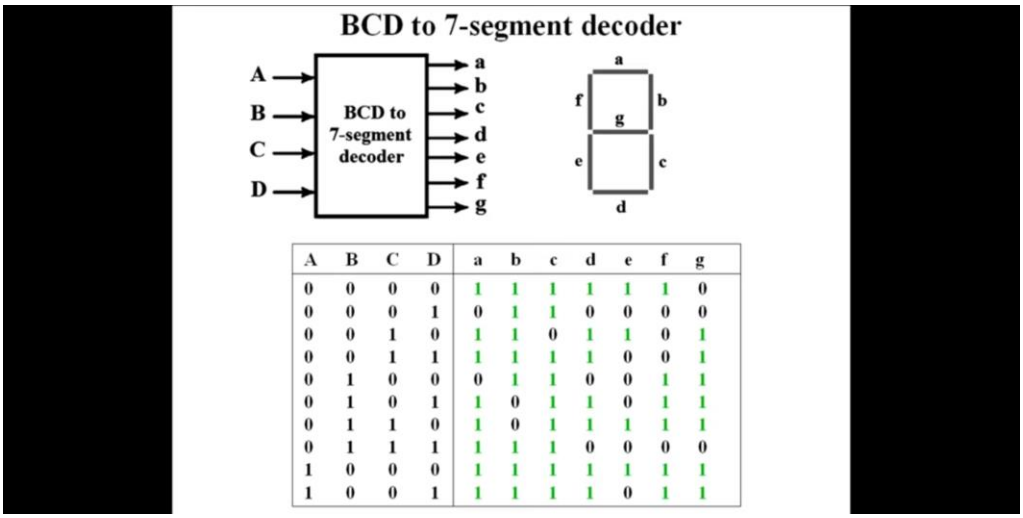
DAILY ASSESSMENT FORMAT

Date:	**28/05/2020**	**Name:**	**Lavanya B**
Course:	**Logic design**	**USN:**	**4al17ec043**
Topic:	**Boolean Algebra** **BCD to 7 segment decoder**	**Semester & Section:**	**6th A**
Github Repository:	**Lavanya-B**		

FORENOON SESSION DETAILS

Image of session





A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Report –

Logic design

29/07/2020

Boolean Algebra is a system of mathematical logic
Set of elements – (0,1)
Two binary operator – OR(+) & AND (·)
unary operator – NOT

AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

A	$\bar{A}$
0	1
1	0

Boolean algebra	ordinary algebra	binary no system
$A+A=A$	$A+A=2A$	$1+1=10$
$1+1=0$	$1+1=2$	$1\cdot 1=1$
$A\cdot A=A$	$A\cdot A=A^2$	
$1\cdot 1=1$	$1\cdot 1=1$	

Axioms (or) postulate.

$x+0=x$ $\quad x+\bar{x}=1$ $\quad x\cdot x=x$
$x+1=1$ $\quad x\cdot 0=0$ $\quad x\cdot\bar{x}=0$
$x+x=x$ $\quad x\cdot 1=x$ $\quad (x)'=x$

The additive identity (OR operation) = 0
The Multiplication identity (And operation) = 1

Laws of Boolean Algebra

1) commutative law:
$x+y=y+x$
$x\cdot y=y\cdot x$

2) Associative Law.
$x+(y+z)=(x+y)+z$
$x\cdot(y\cdot z)=(x\cdot y)\cdot z$

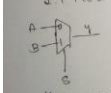
3) Distributive Law.
$x(y+z)=xy+xz$
$x+yz=(x+y)(x+z)$
$=x\cdot x+xz+xy+yz$
$=x+xz+xy+yz$
$=x\{1+z+y\}+yz$
$=x+yz$

Theorems of Boolean algebra

1) Absorption Theorem.
$x+xy=x$, $x+\bar{x}y=x+y$

2)
MUX To logic gate.
1) NAND, NOR – universal gate
2) MUX & Decoder are – universal logic

2:1 MUX

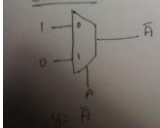


Selection(S)	Output
0	A
1	B

$Y=A\bar{S}+BS$

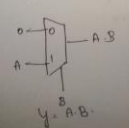
2^n inputs.
n selection line

Inverter



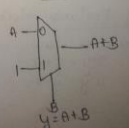
$Y=\bar{A}$

AND

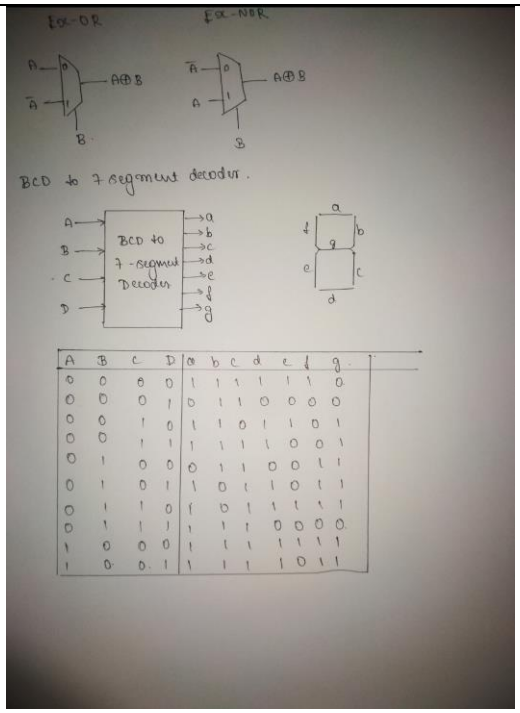


$Y=A\cdot B$

OR



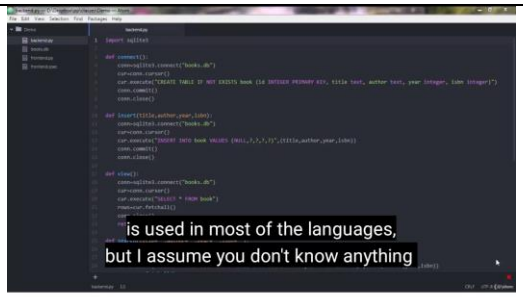
$Y=A+B$



A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Date:	**28/05/2020**	**Name:**	**Lavanya B**
Course:	**Python**	**USN:**	**4al17ec043**
Topic:	**Object Oriented Programming**	**Semester & Section:**	**6th A**

AFTERNOON SESSION DETAILS
Image of session



Lectures More

186 Fixing the Bug (Practice)
Article

187 Solution
Article

188 Creating a Standalone Executable Version of...
Video - 05:00 mins

Section 24 - Object Oriented Programming

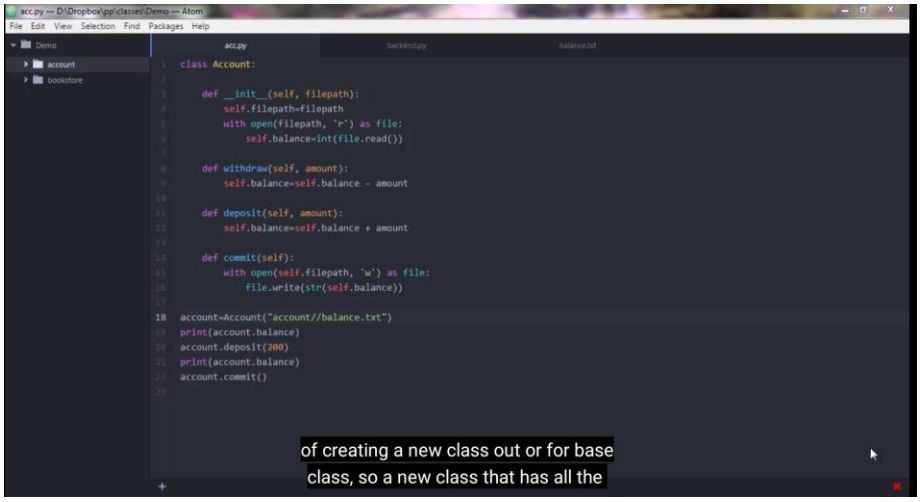
190 Turning this Application into OOP Style, Part 1
Video - 13:01 mins

191 Turning this Application into OOP Style, Part 2
Video - 14:05 mins

192 Creating a Bank Account Object
Video - 21:06 mins

193 Inheritance
Video - 12:08 mins

194 OOP Glossary
Video - 08:12 mins



Report –

Object oriented programming

- Introduction about OOP
- Turning this application intoOOP style
- Creating a bank account object
- Learnt about Inheritance of OOP
- OOP Glossary
- GUI in OOP design

Frontend.py and backend.py script in the OOP style

```
#frontend.py
from tkinter import *
from backend import Database

database=Database("books.db")

class Window(object):

   def __init__(self,window):

      self.window = window

      self.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)

      self.title_text=StringVar()
      self.e1=Entry(window,textvariable=self.title_text)
```

```
        self.e1.grid(row=0,column=1)

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

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

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

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

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

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

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

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

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

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

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

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

        b6=Button(window,text="Close", width=12,command=window.destroy)
```

```
        b6.grid(row=7,column=3)

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

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

    def search_command(self):
        self.list1.delete(0,END)
        for row in
database.search(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_te
xt.get()):
            self.list1.insert(END,row)

    def add_command(self):

database.insert(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_tex
t.get())
        self.list1.delete(0,END)

self.list1.insert(END,(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isb
n_text.get()))

    def delete_command(self):
        database.delete(self.selected_tuple[0])

    def update_command(self):

database.update(self.selected_tuple[0],self.title_text.get(),self.author_text.get(),self.year
_text.get(),self.isbn_text.get())
```

```
window=Tk()
Window(window)
window.mainloop()

#backend.py
import sqlite3
class Database:
   def __init__(self, db):
      self.conn=sqlite3.connect(db)
      self.cur=self.conn.cursor()
      self.cur.execute("CREATE TABLE IF NOT EXISTS book (id INTEGER PRIMARY KEY, title
text, author text, year integer, isbn integer)")
      self.conn.commit()
   def insert(self,title,author,year,isbn):
      self.cur.execute("INSERT INTO book VALUES (NULL,?,?,?,?)",(title,author,year,isbn))
      self.conn.commit()
   def view(self):
      self.cur.execute("SELECT * FROM book")
      rows=self.cur.fetchall()
      return rows
   def search(self,title="",author="",year="",isbn=""):
      self.cur.execute("SELECT * FROM book WHERE title=? OR author=? OR year=? OR
isbn=?", (title,author,year,isbn))
      rows=self.cur.fetchall()
      return rows
   def delete(self,id):
      self.cur.execute("DELETE FROM book WHERE id=?",(id,))
      self.conn.commit()
   def update(self,id,title,author,year,isbn):
      self.cur.execute("UPDATE book SET title=?, author=?, year=?, isbn=? WHERE
id=?",(title,author,year,isbn,id))
      self.conn.commit()
   def __del__(self):
      self.conn.close()
```