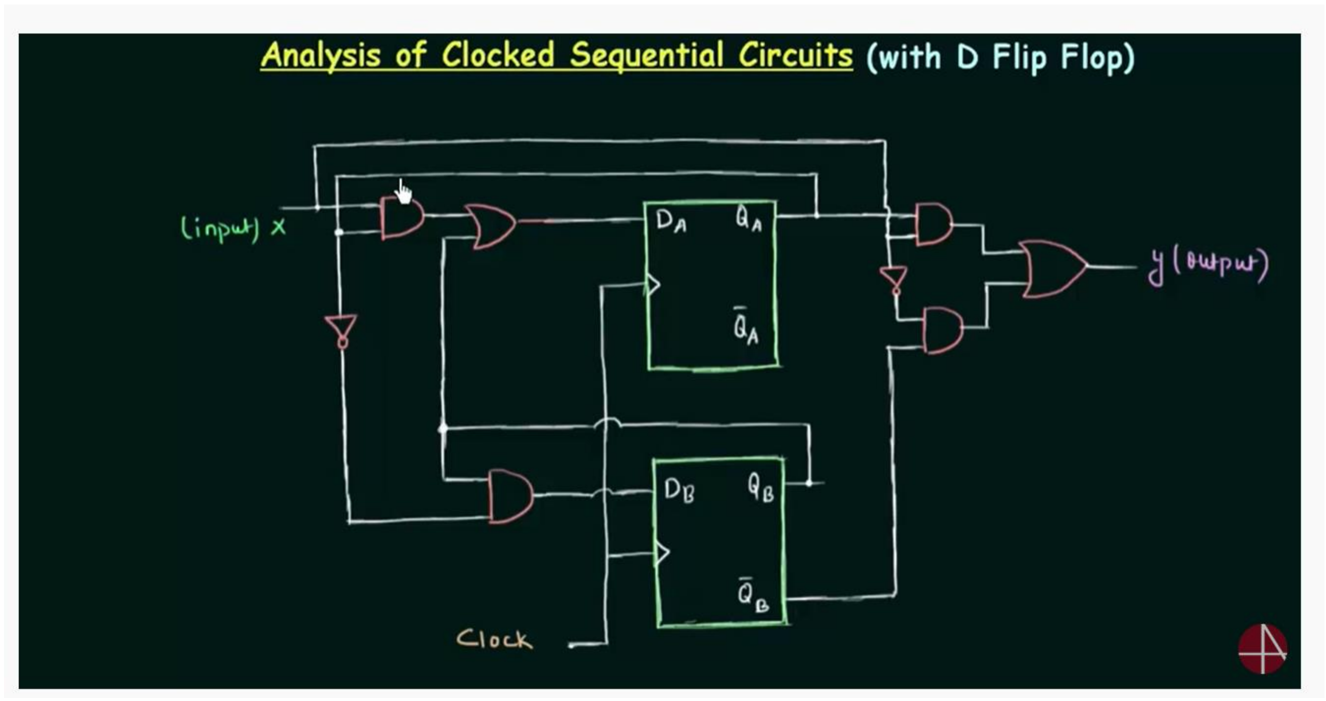
DAILY ASSESSMENT FORMAT

Date:	**29 MAY 2020**	**Name:**	**PAVITHRAN S**
Course:	**LOGIC DESIGN**	**USN:**	**4AL17EC068**
Topic:	**Analysis of clocked sequential circuits** **Digital clock design**	**Semester & Section:**	**6th B**
Github Repository:	**Pavithran**		

FORENOON SESSION DETAILS
Image of session  Analysis of Clocked Sequential Circuits (with D Flip Flop) (input) x D_A Q_A Q̄_A D_B Q_B Q̄_B Clock y (output)

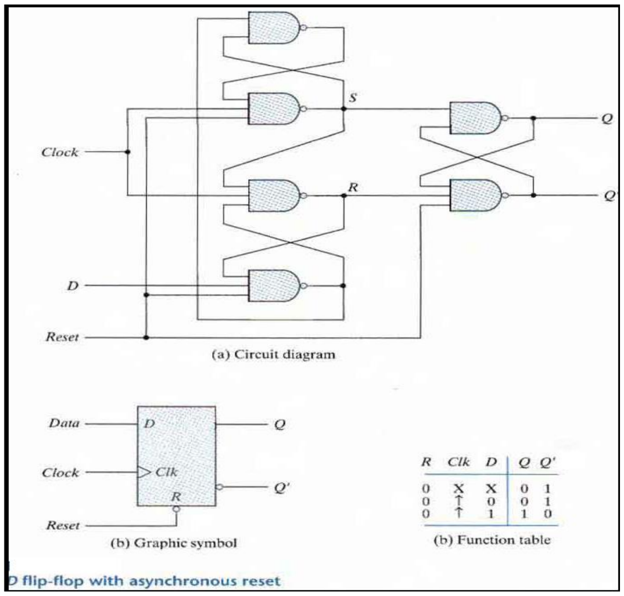
Report – Report can be typed or hand written for up to two pages.

ANALYSIS OF CLOCKED SEQUENTIAL CIRCUITS

- Some flip-flops have asynchronous inputs that are used to force the flip-flop to a particular state independently of the clock
- The input that sets the **flip-flop to 1 is called preset or direct set.** The input that clears the flip-flop to 0 is called **clear or direct reset.**
- When power is turned on in a digital system, the state of the flip-flops is unknown. The direct inputs are useful for bringing all flip-flops in the system to a known starting state prior to the clocked operation.
- The knowledge of the type of flip-flops and a list of the Boolean expressions of the combinational circuit provide the information needed to draw the logic diagram of the sequential circuit. The part of the combinational circuit that gene rates external outputs is described algebraically by a set of Boolean functions called **output equations.** The part of the circuit that generates the inputs to flip-flops is described algebraically by a set of Boolean functions called flip-flop input equations **(or excitation equations).**
- The information available in a state table can be represented graphically in the form of a **state diagram.** In this type of diagram a state is represented by a circle and the (clock-triggered) transitions between states are indicated by directed lines connecting the circles.
- The time sequence of inputs, outputs, and flip-flop states can be enumerated in a state table (transition table). The table has four parts present state, next state, inputs and outputs.
- In general a sequential circuit with **'m' flip-flops and 'n' inputs needs 2^{m+n}** rows in the state table.

Positive Edge Triggered D Flip-flop

- A circuit diagram of a Positive edge triggered D Flip-flop is shown as below. It has an **additional reset input** connected to the three NAND gates.



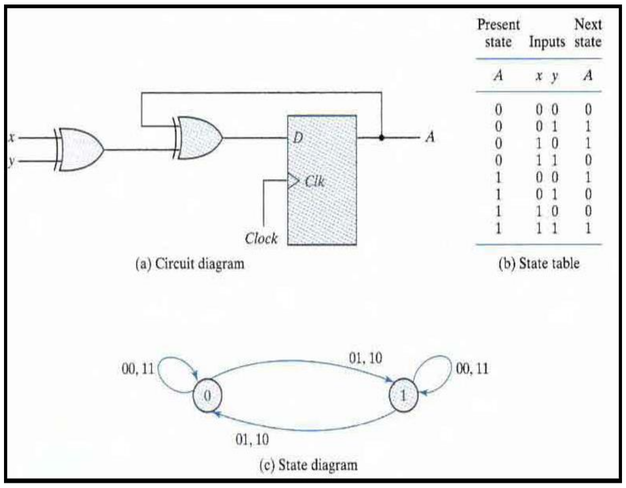
R	Clk	D	Q	Q'
0	X	X	0	1
0	↑	0	0	1
0	↑	1	1	0

- When the **reset input is 0 it forces output Q' to Stay at 1** which clears output Q to 0 thus resetting the flip-flop.
- Two other connections from the reset input ensure that the S input of the third **SR latch stays at logic 1** while the reset input is at 0 regardless of the values of D and Clk.
- Function table suggests that:
    - **When R = 0, the output is set to 0 (independent of D and Clk).**
    - The clock at Clk is shown with an upward arrow to indicate that the flip-flop triggers on the positive edge of the clock.
    - The value in D is transferred to Q with every positive-edge clock signal provided that R = 1.

Analysis with D Flip-Flops
The input equation of a D Flip-flop is given by **$D_A = A \oplus x \oplus y$.** D_A means a D Flip-flop with output A.
The x and y variables are the inputs to the circuit. No output equations are given, which implies that the output comes from the output of the flip-flop.

The state table has one column for the present state of flip-flop 'A' two columns for the two inputs, and one column for the next state of A.

- The next-state values are obtained from the state equation **$A(t + 1) = A \oplus x \oplus y$.**
- The expression specifies an odd function and is equal to 1 when only one variable is 1 or when all three variables are 1.



Present state	Inputs	Next state
A	x y	A
0	0 0	0
0	0 1	1
0	1 0	1
0	1 1	0
1	0 0	1
1	0 1	0
1	1 0	0
1	1 1	1

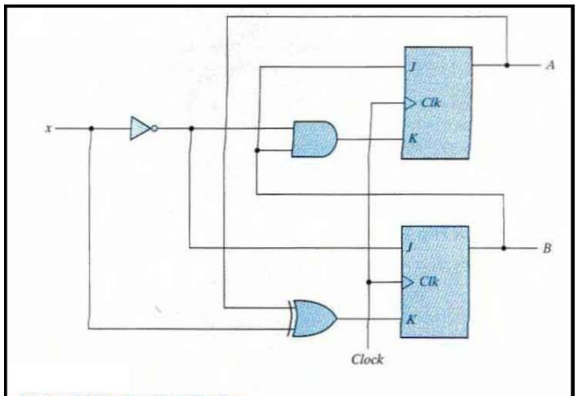
(b) State table

Analysis with JK Flip-Flops

- The circuit can be specified by the flip-flop input equations:
    - **$J_A = B$; $K_A = Bx'$**

    - **$J_B = x'$; $K_B = A'x + Ax' = A \oplus x$**

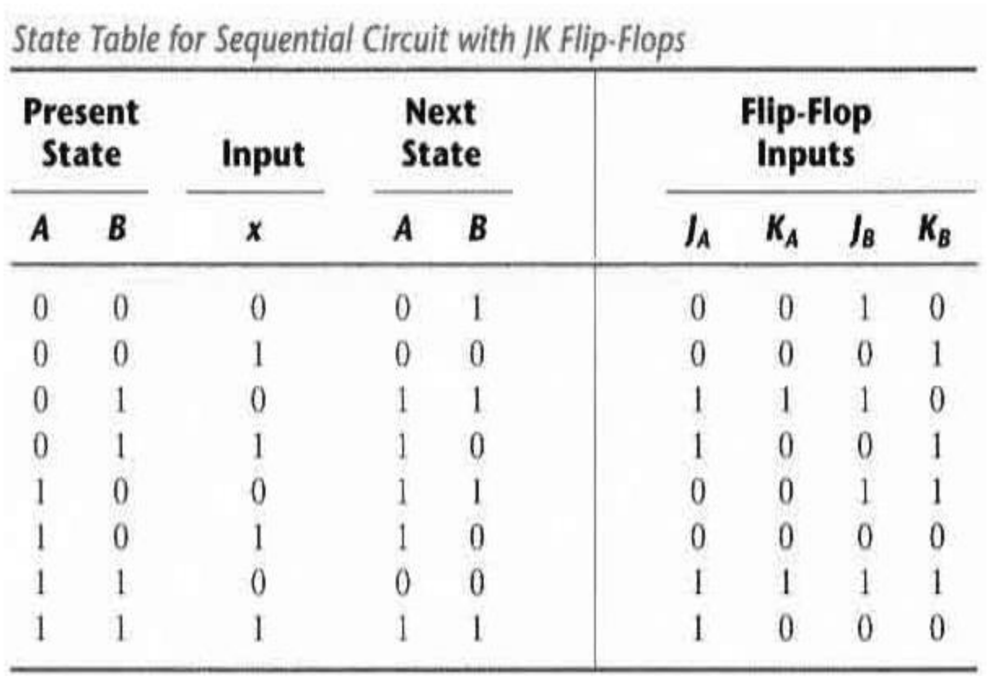
- The next state of each flip-flop is evaluated from the corresponding J and K inputs and the characteristic table of the JK flip-flop listed as:
    - **When J = 1 and K = 0 the next state is 1**
    - **When J = 0 and K = 1 the next state is 0**
    - **When J = 0 and K = 0 there is no change of state and the next-state value is the same as that of the present state.**
    - **When J = K = 1, the next-state bit is the complement of the present-state bit.**



. Sequential circuit with *JK* flip-flop

- The characteristic equations for the flip-flops are
  - **$A(t + 1) = JA' + K'A$**
  - **$B(t + 1) = JB' + K'B$**

- This gives us the state equation of A by substituting the values of J_A, K_A



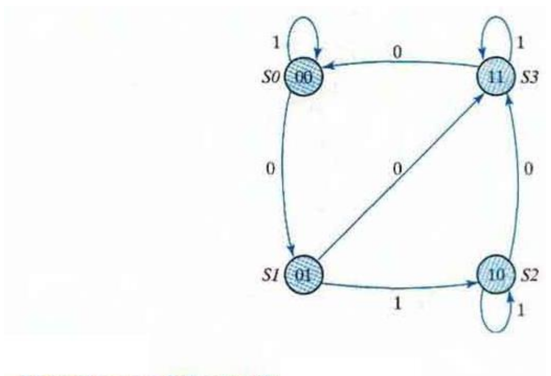
State Table for Sequential Circuit with JK Flip-Flops

Present State		Input	Next State		Flip-Flop Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

  - **$A(t + 1) = BA' + (Bx')'A = A'B + AB' + Ax$**

- The state equation provides the bit values for the column headed "Next State" for A in the state table. Similarly, the state equation for flip-flop B can be derived from the characteristic equation by substituting the values of J_B and K_B.:

  - **$B(t + 1) = x'B' + (A \oplus x)'B = B'x' + ABx + A'Bx'$**



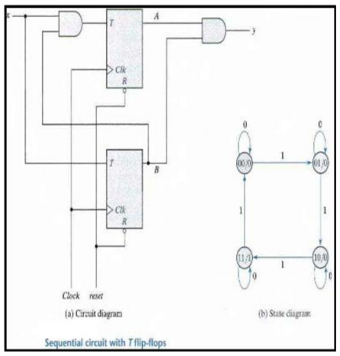
State diagram of the circuit

Analysis with T Flip-Flops

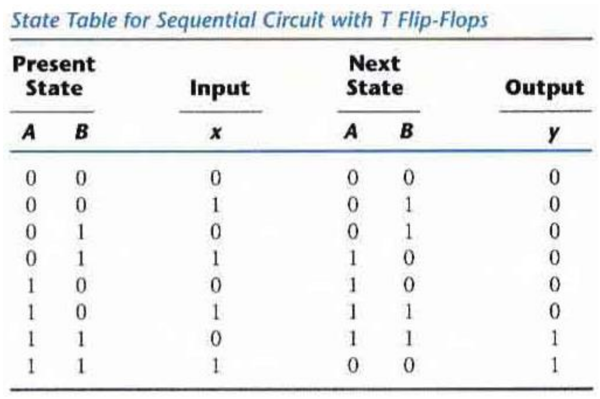
- The circuit can be specified by the characteristic equations:
    - **$Q(t+1) = T \oplus Q = T'Q + TQ'$**

- The sequential circuit has two flip-flops A and B, one input x, and one output y and can be described algebraically by two input equations and an output equation:
    - **$T_A = Bx$**
    - **$T_B = x$**
    - **$y = AB$**

- The state table for the circuit is listed below. The values for y are obtained from the output equation. The values for the next state can be derived from the state equations by substituting T_A and T_B in the characteristic equations yielding:
    - **$A(t + 1) = (Bx)' A + (Bx)A' = AB' + Ax' + A'Bx$**
    - **$B(t + 1) = x \oplus B$**



Sequential circuit with T flip-flops



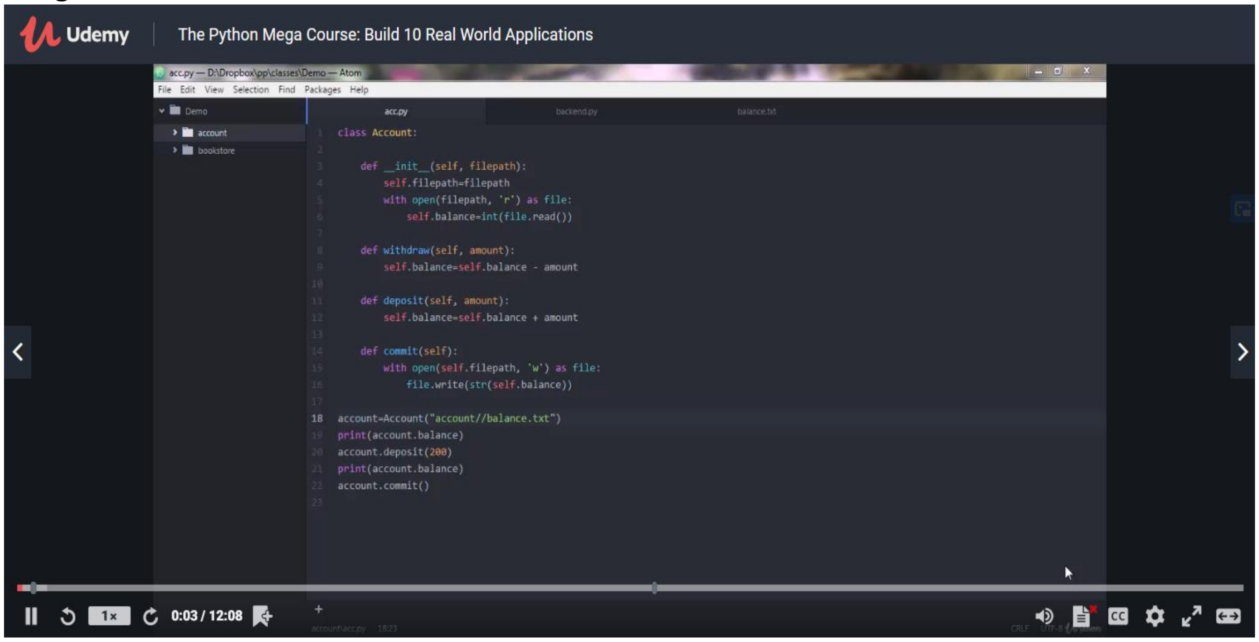
State Table for Sequential Circuit with T Flip-Flops

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

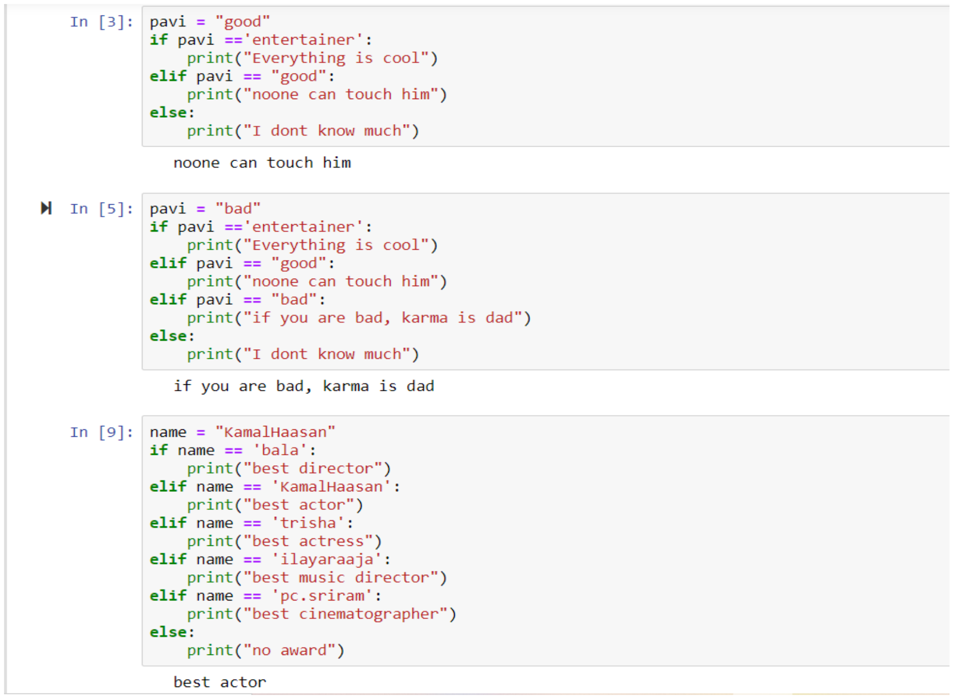
Date:	29 MAY 2020	Name:	PAVITHRAN S
Course:	PYTHON	USN:	4AL17EC068
Topic:	Chaining comparision operators, conditional statements	Semester & Section:	6th B
Github Repository:	Pavithran		

AFTERNOON SESSION DETAILS

Image of session


Udemy
The Python Mega Course: Build 10 Real World Applications
acc.py — D:\Dropbox\pp\classes\Demo — Atom
File Edit View Selection Find Packages Help
Demo
account
bookstore
acc.py
backend.py
balance.txt
0:03 / 12:08
1×

```python
class Account:

    def __init__(self, filepath):
        self.filepath=filepath
        with open(filepath, 'r') as file:
            self.balance=int(file.read())

    def withdraw(self, amount):
        self.balance=self.balance - amount

    def deposit(self, amount):
        self.balance=self.balance + amount

    def commit(self):
        with open(self.filepath, 'w') as file:
            file.write(str(self.balance))

account=Account("account//balance.txt")
print(account.balance)
account.deposit(200)
print(account.balance)
account.commit()
```

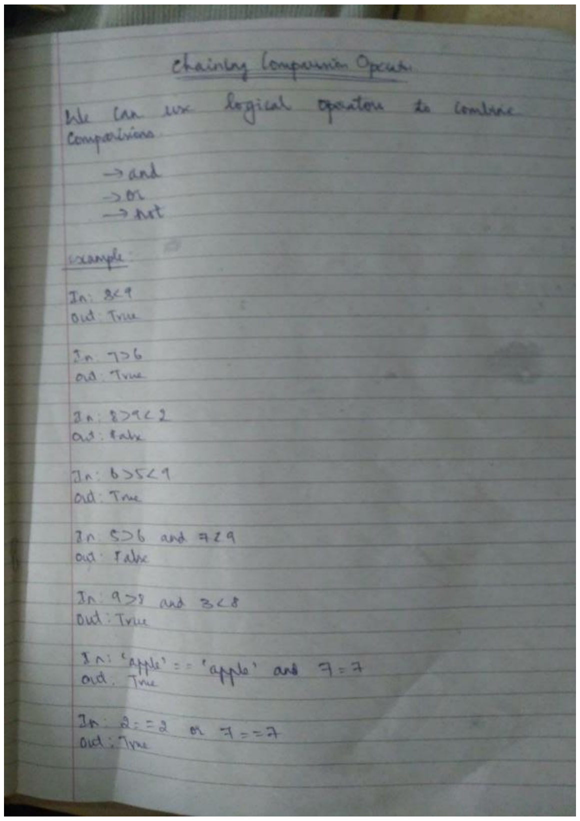


```
In [3]: pavi = "good"
        if pavi =='entertainer':
            print("Everything is cool")
        elif pavi == "good":
            print("noone can touch him")
        else:
            print("I dont know much")

           noone can touch him
```

```
In [5]: pavi = "bad"
        if pavi =='entertainer':
            print("Everything is cool")
        elif pavi == "good":
            print("noone can touch him")
        elif pavi == "bad":
            print("if you are bad, karma is dad")
        else:
            print("I dont know much")

           if you are bad, karma is dad
```

```
In [9]: name = "KamalHaasan"
        if name == 'bala':
            print("best director")
        elif name == 'KamalHaasan':
            print("best actor")
        elif name == 'trisha':
            print("best actress")
        elif name == 'ilayaraaja':
            print("best music director")
        elif name == 'pc.sriram':
            print("best cinematographer")
        else:
            print("no award")

           best actor
```

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



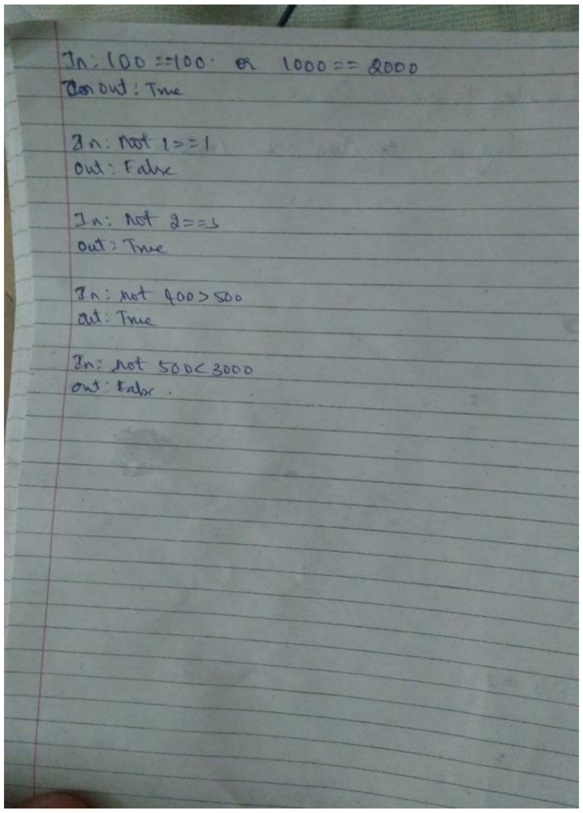
Chaining Comparison Operators

We can use logical operators to combine comparisons

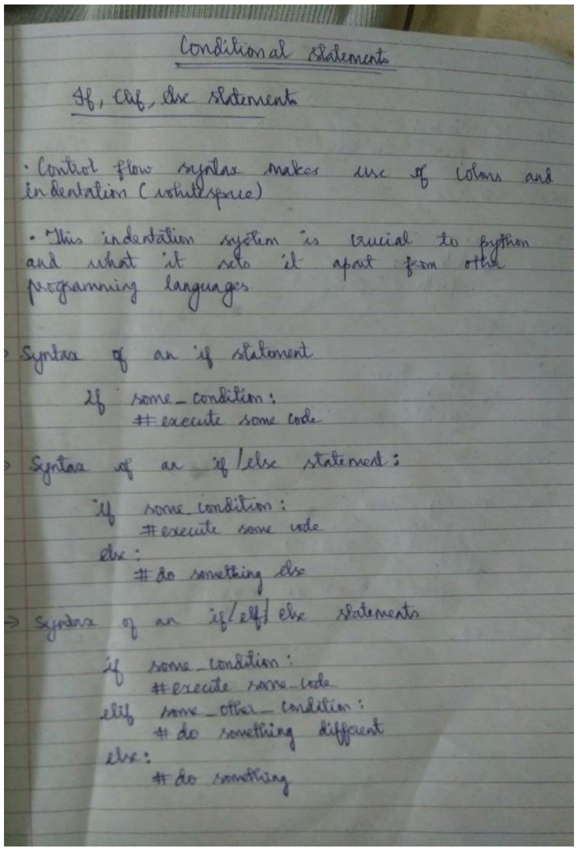
→ and
→ or
→ not

example

```
In: 8<9
Out: True

In: 7>6
Out: True

In: 8>9<2
Out: False

In: 6>5<9
Out: True

In: 5>6 and 7<9
Out: False

In: 9>8 and 3<8
Out: True

In: 'apple' == 'apple' and 7=7
Out: True

In: 2==2 or 7==7
Out: True
```



```
In: 100==100 or 1000==2000
~~Ou~~ Out: True

In: not 1==1
Out: False

In: not 2==3
Out: True

In: not 400>500
Out: True

In: not 500<3000
Out: False .
```



Conditional Statements

If, elif, else statements

• Control flow syntax makes use of colons and indentation (whitespace)

• This indentation system is crucial to python and what it sets it apart from other programming languages

→ Syntax of an if statement

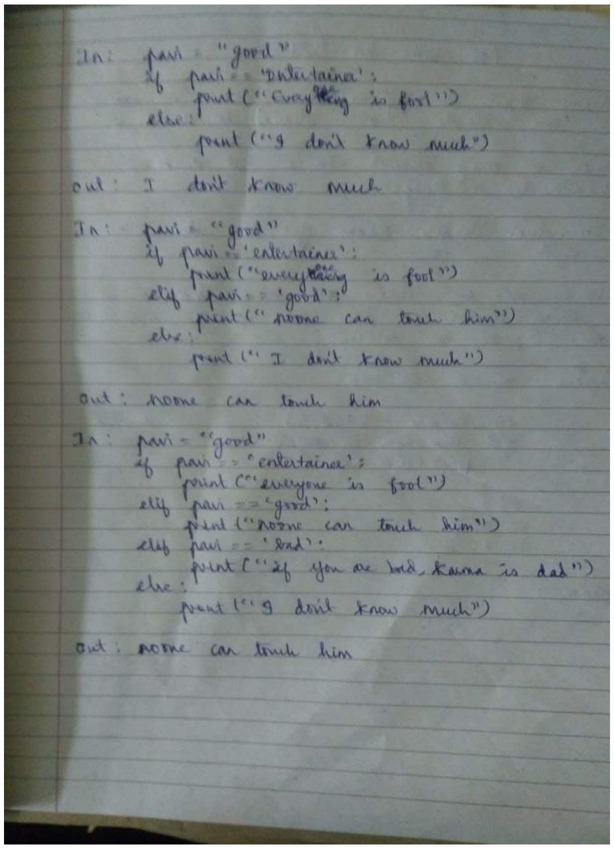
```
if some_condition:
    #execute some code
```

→ Syntax of an if/else statement:

```
if some_condition:
    #execute some code
else:
    # do something else
```

→ Syntax of an if/elif/else statements

```
if some_condition:
    #execute some code
elif some_other_condition:
    # do something different
else:
    # do something
```



```
In:  pavi = "good"
     if pavi == 'Entertainer':
         print ("Everything is fool")
     else:
         print ("I don't know much")

Out: I don't know much

In:  pavi = "good"
     if pavi == 'entertainer':
         print ("everyone is fool")
     elif pavi == 'good':
         print ("noone can touch him")
     else:
         print (" I don't know much")

Out: noone can touch him

In:  pavi = "good"
     if pavi == 'entertainer':
         print ("everyone is fool")
     elif pavi == 'good':
         print ("noone can touch him")
     elif pavi == 'bad':
         print ("if you are bad, Karma is dad")
     else:
         print ("I don't know much")

Out: noone can touch him
```