Bazla Bilquees
U3312671
Introduction To Information Technology

ASSIGNMENT # 02 – PYTHON FUNDAMENTAL

Case Study 3 – A Boolean Circuit Equivalence

Step 1 — Understand the Problem:

We receive Boolean circuits and have to model their logic as Boolean expressions, which must be performed in Python. Lastly, we check that the Python code is acting properly under all input combinations.

Step 2 — Inputs & Outputs

Circuit a)
Inputs: A, B, C (True/False or 0/1)
Output: Y (Boolean: True/FALSE or 0/1)
Circuit b)
Inputs: P, Q, R (Boolean: True/False or 0/1)
Output: Z (Boolean: True/False or 0/1).
Units: All are Boolean values; Python will use True/False.

Step 3 — Algorithm

Steps for Circuit a:

- Read inputs A, B, C.
- Use logic gates in a series (e.g. AND, OR, NOT) to obtain an intermediate result.
- Use the output of the intermediate results based on the Boolean expression to generate the output Y.
- Return/print Y.

Steps for Circuit b:

- Read inputs P, Q, R.
- Use logic gates to find intermediate results.
- Add the results to obtain output Z.
- Return/print Z.

Reference I/O: The input variables of each step generate intermediate Boolean values to the final output.

Step 4 — Flow Chart

u3312671-Bazla Bilquees- Assignment

Step 4 – PSEUDOCODE

Circuit a:
START
INPUT A, B, C
X1 = NOT A
X2 = B AND C
Y = X1 OR X2
OUTPUT Y
END
Circuit b:
START
INPUT P, Q, R
X1 = P OR NOT Q
X2 = Q AND R
Z = X1 AND X2
OUTPUT Z
END

Step 5 Python Code

For Circuit a:

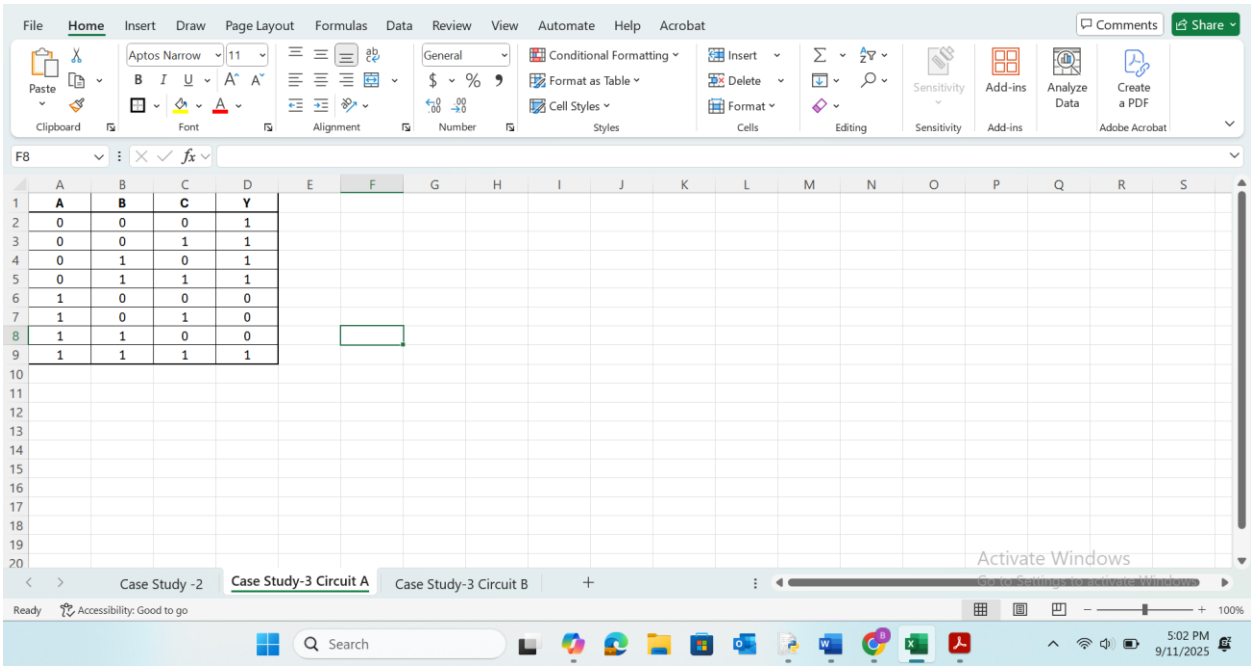
```
def circuit_a(A, B, C):
    X1 = not A
    X2 = B and C
    Y = X1 or X2
    return Y
# Test all combinations
for A in [True, False]:
    for B in [True, False]:
        for C in [True, False]:
            print(f"A={A}, B={B}, C={C} => Y={circuit_a(A,B,C)}")
```

For Circuit b:

```
def circuit_b(P, Q, R):
    X1 = P or not Q
    X2 = Q and R
    Z = X1 and X2
    return Z
# Test all combinations
for P in [True, False]:
    for Q in [True, False]:
        for R in [True, False]:
            print(f"P={P}, Q={Q}, R={R} => Z={circuit_b(P,Q,R)}")
```

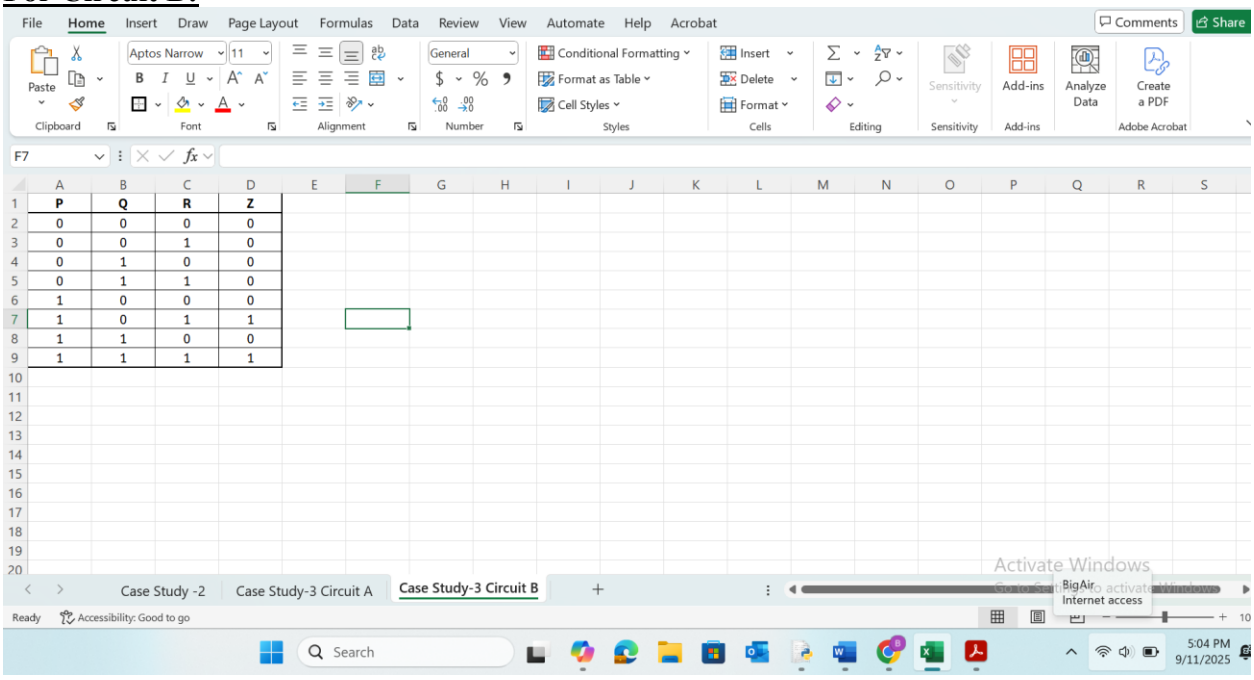
Step 6 -Testing: handwritten expected results + test runs & notes

For Circuit A:



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

For Circuit B:



P	Q	R	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

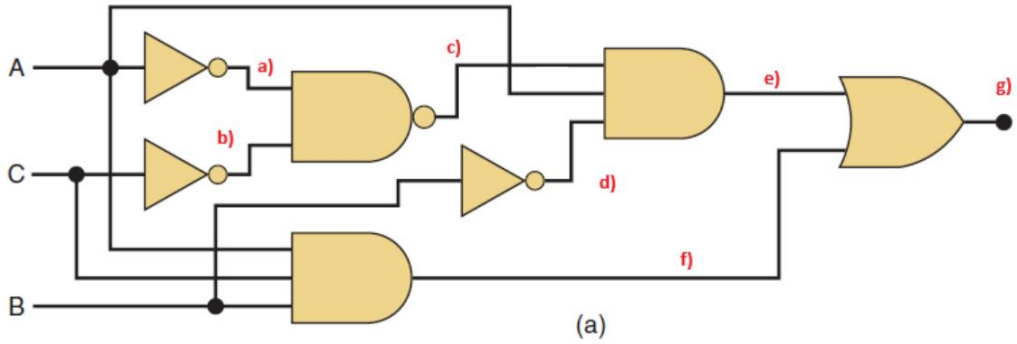
```
Exiting Smart Classroom Monitor. Goodbye!
>>>
========== RESTART: C:/Users/Lenovo/Desktop/Case 3.py ==========
A=True, B=True, C=True => Y=True
A=True, B=True, C=False => Y=False
A=True, B=False, C=True => Y=False
A=True, B=False, C=False => Y=False
A=False, B=True, C=True => Y=True
A=False, B=True, C=False => Y=True
A=False, B=False, C=True => Y=True
A=False, B=False, C=False => Y=True
>>>
========== RESTART: C:/Users/Lenovo/Desktop/case study 3- circuit b.py ==========
P=True, Q=True, R=True => Z=True
P=True, Q=True, R=False => Z=False
P=True, Q=False, R=True => Z=False
P=True, Q=False, R=False => Z=False
P=False, Q=True, R=True => Z=False
P=False, Q=True, R=False => Z=False
P=False, Q=False, R=True => Z=False
P=False, Q=False, R=False => Z=False
>>>
```


Ln: 188 Col: 0

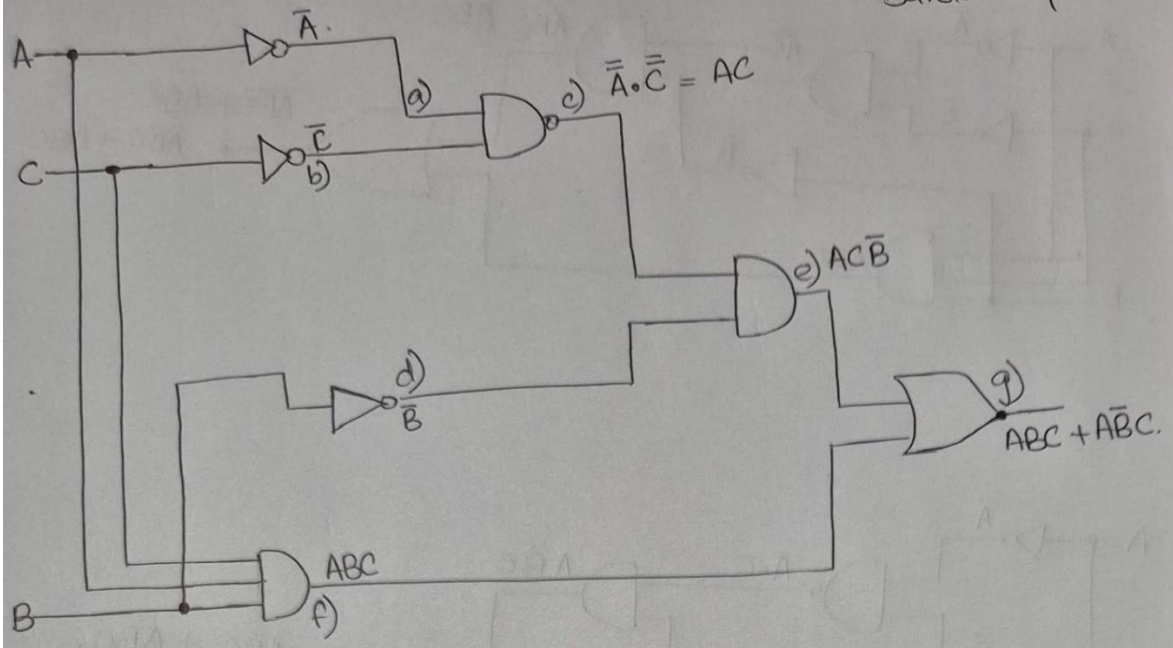
Search 5:04 PM 9/11/2025

Step 7- Boolean expression:

For Circuit A:



U3312671
Bazla Bilqees



a) A $\quad \bar{A}$

b) C $\quad \bar{C}$

c) $\bar{A}.\bar{C} \quad \bar{\bar{A}}.\bar{\bar{C}} = AC.$

d) B $\quad \bar{B}$

e) $AC\bar{B} \quad A\bar{B}C$

f) ABC $\quad A.B.C$

g) e+f $\quad ABC + A\bar{B}C.$

$\therefore ABC + A\bar{B}C$

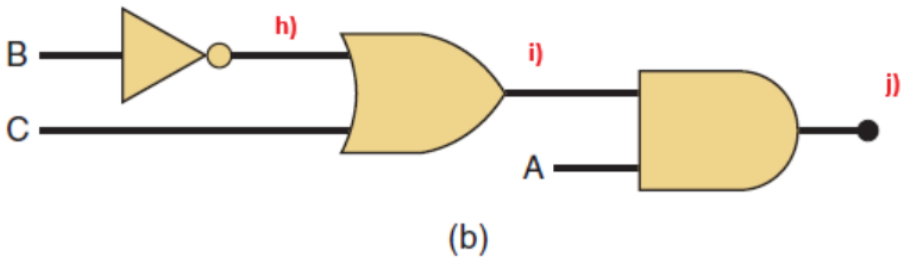
$AC(B+\bar{B}).$

$\therefore B+\bar{B} = 1.$

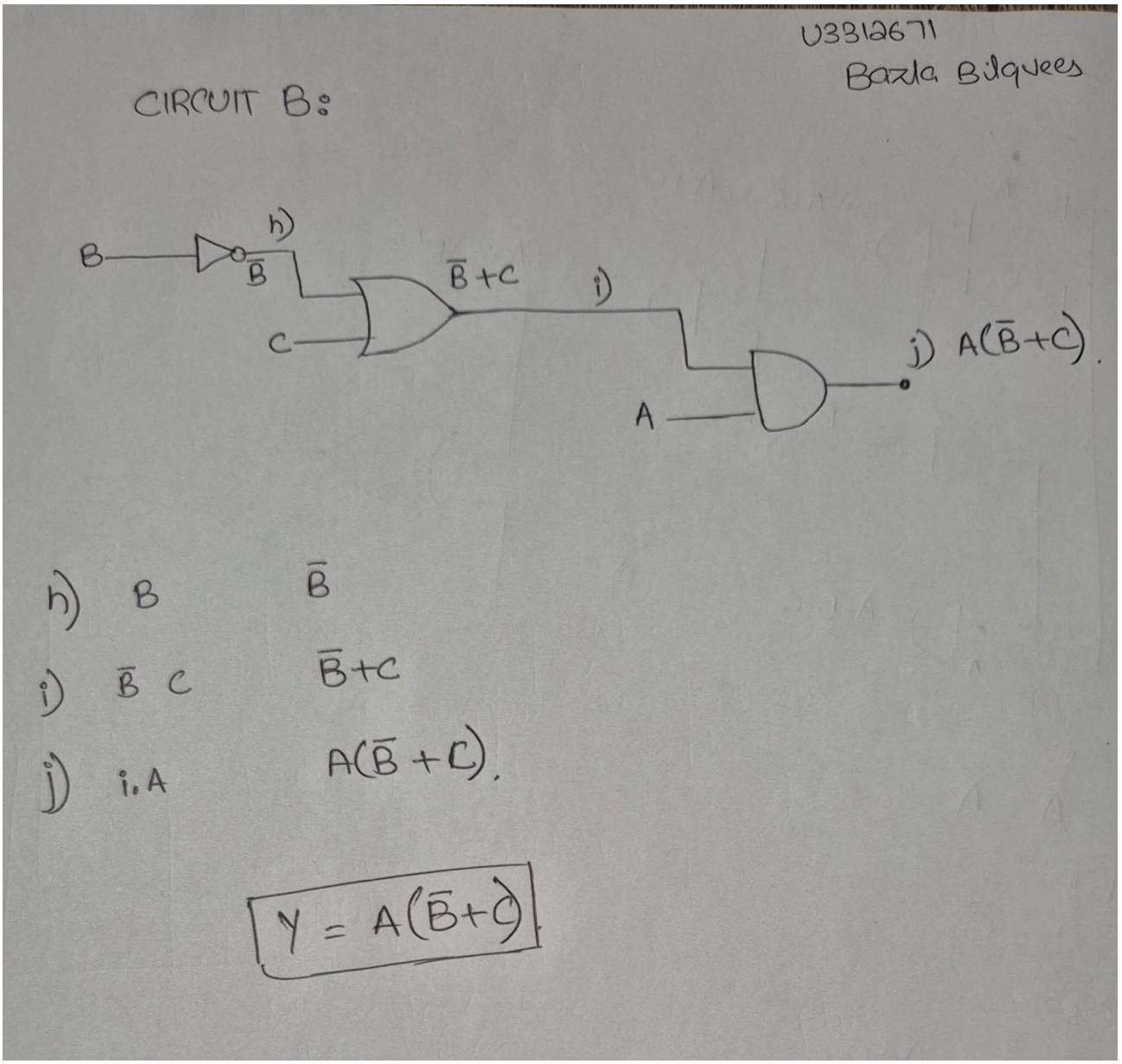
$\boxed{X = AC}$

So, the Boolean expression is **X = AC.**

For Circuit B:



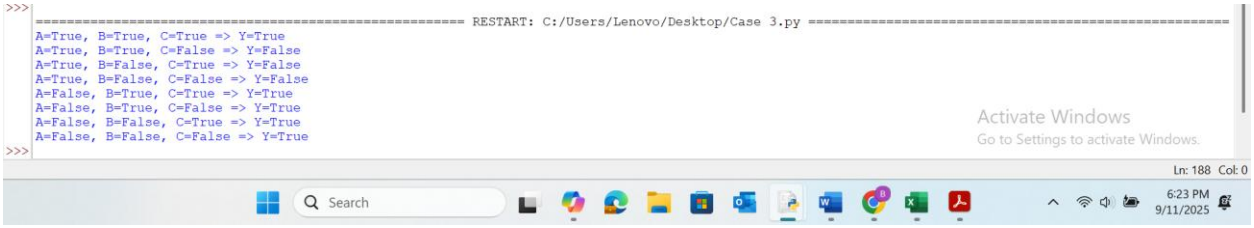
(b)



Step 8 – Truth Table of Both Circuits:

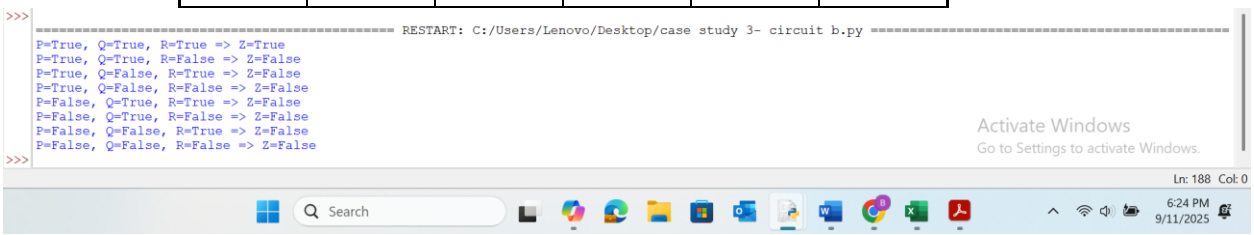
Circuit A:

A	B	C	a= NOT A	b= NOT C	c = NOT A AND NOT C	d = NOT B	e = c AND d	f = AND ABC	g = e OR f
0	0	0	1	1	1	1	1	0	1
0	0	1	1	0	0	1	0	0	0
0	1	0	1	1	1	0	0	0	0
0	1	1	1	0	0	0	0	0	0
1	0	0	0	1	0	1	0	0	0
1	0	1	0	0	0	1	0	0	0
1	1	0	0	1	0	0	0	0	0
1	1	1	0	0	0	0	0	1	0

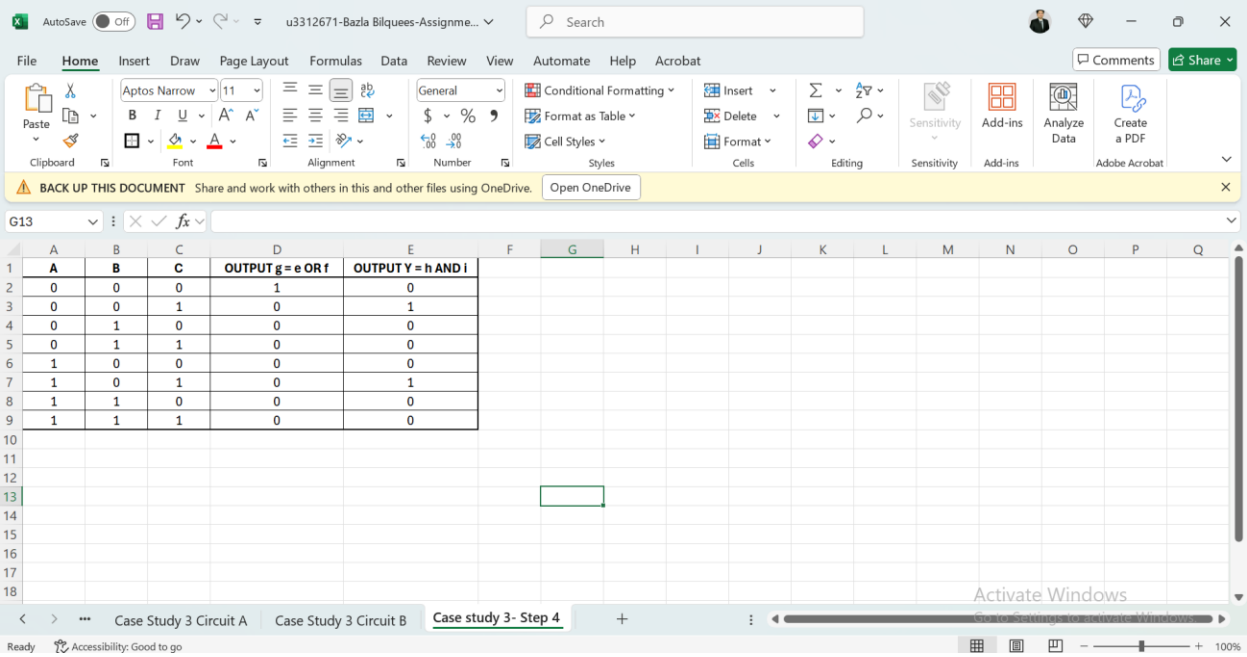


Circuit B:

A	B	C	NOT B (h)	B OR C (i)	OUTPUT Y = h AND i
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	1	1
1	1	0	0	1	0
1	1	1	0	1	0



Step 9 - Check that both circuits are equivalent.



A	B	C	OUTPUT g = e OR f	OUTPUT Y = h AND i
0	0	0	1	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	1
1	1	0	0	0
1	1	1	0	0

Both Circuit are not equivalent to each other

```
# Each element is a tuple (X, Y)
truth_a = [
    (0,0),  # A=0 B=0 C=0
    (1,0),  # A=0 B=0 C=1
    (0,1),
    (1,1),
    (0,0),
    (1,0),
    (0,1),  # example row 110
    (1,1),
]
truth_b = [
    (0,0),
    (1,0),
    (0,1),
    (1,1),
    (0,0),
    (1,0),
    (1,1),  # differs from truth_a at row index 6
    (1,1),
]

assert len(truth_a) == len(truth_b), "Both truth tables must have same length"
diffs = []
for i, (out_a, out_b) in enumerate(zip(truth_a, truth_b)):
```

```
    A = (i >> 2) & 1
    B = (i >> 1) & 1
    C = i & 1
    if out_a != out_b:
        diffs.append((i, A, B, C, out_a, out_b))
if not diffs:
    print("Equivalent: all outputs match for every input.")
else:
    print("NOT equivalent. Differences found:")
    for idx, A, B, C, out_a, out_b in diffs:
        print(f" Row {idx}: A={A} B={B} C={C}  CircuitA={out_a}  CircuitB={out_b}")
```