

# **ASSIGNMENT # 02 – PYTHON FUNDAMENTAL**

## **CASE STUDY # 01 CAMPUS CAFÉ CHECKOUT**

### **Step 1 — Understand the Problem:**

Develop a console point-of-sale application that allows an individual to navigate a menu at a cafe, add items to a collections cart, view the items in the cart and check out. Line items, subtotal, tax up to 10% and optional student discount 5% and a final total must be displayed on the receipt. Use a dictionary to use the menu, a list to use the cart and a set to use categories.

### **Step 2 — Inputs & Outputs**

<b>SYMBOL</b>	<b>Name</b>	<b>Type</b>	<b>Role</b>	<b>Description / logic</b>
F	Food present	Boolean	Input	1 if cart contains at least one <b>food</b> item, else 0
D	Drink present	Boolean	Input	1 if cart contains at least one <b>drink</b> item, else 0
S	Student flag	Boolean	Input	1 if user confirms they are a student (student discount applies), else 0
P	Cart non-empty	Boolean	Input	1 if cart has at least 1 item, else 0
M	Meal-deal	Boolean	Output	1 if meal-deal discount applies (food AND drink present)
SD	Student Discount	Boolean	Output	1 if student discount applies (S)
CD	Combined Discount Flag	Boolean	Output	1 if <i>any</i> discount is available (meal-deal OR student)
CO	Checkout allowed	Boolean	Output	1 if checkout can proceed (cart non-empty = P)
PR	Print Receipt	Boolean	Output	1 if CO = 1 (receipt should be printed after checkout)

## Step 3 — Algorithm

1. Start
2. Enter menu items, prices and categories (Drink or Food).
3. Initialize an empty cart.
4. Continue till one decides to quit:
  - 4.1 Display main menu:
    1. Show Menu
    2. Add Item
    3. View Cart
    4. Checkout
    5. Exit
  - 4.2 Read user choice
  - 4.3 If choice = Show Menu:
    - List all menu items including price and category.
  - 4.4 If choice = Add Item:
    - Ask user to enter item name
    - Check if item exists in menu
    - Yes, request quantity (default= 1)
    - Add (item, quantity) to cart
    - Display confirmation
    - If no, display "Item not on menu"
  - 4.5 If choice = View Cart:
    - In case there is nothing in the cart, list "Cart is empty"
    - otherwise, list everything in cart, including quantity and total price.
  - 4.6 If choice = Checkout:
    - In case of cart empty, it will show Nothing to checkout and move on.
    - Subtotal = (price x amount) of all items in cart.
    - Calculate tax = subtotal × 10%

4.7 If choice = Exit:

- Display "Exiting program"
- Stop loop

4.8 Else:

- Display "Invalid choice"

End program

## Step 4 — Flow Chart



u3312671-bazla  
Bilquees- Assignment

## Step 4 – PSEUDOCODE

```
BEGIN
    define menu items including price and category.
    MENU = {
        "Coffee": (3.50, "Drink"),
        "Tea": (2.50, "Drink"),
        "Muffin": (2.00, "Food"),
        "Sandwich": (5.00, "Food"),
        "Smoothie": (4.00, "Drink"),
        "Salad": (4.50, "Food")
    }
    CART = []
    FUNCTION show_menu()
        PRINT "--- Café Menu ---"
        FOR each ITEM in MENU
            Name, price, category of PRINT ITEM.
        END FOR
    END FUNCTION
    FUNCTION add_item(CART)
        CALL show_menu()
        INPUT item_name
        IF itemname in MENU THEN
            INPUT quantity (default 1)
            ADD (name of item, amount) to CART.
            Print x number of item-name added to cart.
        ELSE
            PRINT "Item not on the menu"
        END IF
    END FUNCTION
```

```

FUNCTION view_cart(CART)
IF CART is empty THEN
PRINT "Cart is empty"
RETURN
END IF
PRINT "--- Current Cart ---"
Qty in CART FOR each item.
price = MENU[item].price
PRINT quantity x item - price*quantity
END FOR
END FUNCTION

FUNCTION checkout(CART)
IF CART is empty THEN
PRINT "Cart is empty. Nothing to checkout."
RETURN
END IF
subtotal = The total of (price x quantity) of items in CART.
tax = subtotal * 0.10

// Student discount
GRAB apply student discount (y/n).
IF yes THEN
discount = (subtotal + tax) * 0.05
ELSE
discount = 0
END IF

Promotion: 2 drink discount and food combo: 2 drinks off and food combo.
drinks count = SUM of the quantity of items in the CART whose category is a drink.
foods count = SUM of quantities of items that are in CART and category = Food.
combos = minimum of drinks, foods count

```

```

meal_deal_discount = combos * 2.00
subtotal = total, less tax, less discount, less meal deal discount.

// Print receipt
PRINT "--- Receipt ---"
Qty in CART FOR each item.
PRINT quantity x item - price*quantity
END FOR
PRINT subtotal, tax
IF discount > 0 then PRINT student discount -discount.
IF meal_deal_discount >0 then PRINT meal_deal_discount -meal_deal_discount.
PRINT total
PRINT "Thank you for visiting!"
END FUNCTION

FUNCTION main()
WHILE True
PRINT menu choices: Show Menu, Add Item, View Cart, Checkout, Exit
INPUT user_choice
SWITCH user_choice
CASE 1: CALL show_menu()
CASE 2: CALL add_item(CART)
CASE 3: CALL view_cart(CART)
CASE 4: CALL out(CART), and CLEAR CART.
CASE 5: PRINT "Leaving program" BREAK
DEFAULT: PRINT "Illegal option"
END SWITCH
END WHILE
END FUNCTION
ALL main()
END

```

## Step 5 Truth Table

F	D	K	S	T	P	N	MULTICAT	MEAL	DISC	RECEIPT	CHECKOUT_OK
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	1	0
0	0	1	0	0	1	1	0	0	0	1	1
0	0	1	0	1	0	1	0	0	0	1	0
0	0	1	0	1	1	1	0	0	0	1	1
0	0	1	1	0	0	1	0	0	1	1	0
0	0	1	1	0	1	1	0	0	1	1	1
0	0	1	1	1	0	1	0	0	0	1	0
0	0	1	1	1	1	1	0	0	1	1	1
0	1	0	0	0	0	1	0	0	0	1	0
0	1	0	0	0	1	1	0	0	0	1	1
0	1	0	0	1	0	1	0	0	0	1	0
0	1	0	0	1	1	1	0	0	0	1	1
0	1	0	1	0	0	1	0	0	1	1	0
0	1	0	1	0	1	1	0	0	1	1	1
0	1	0	1	1	0	1	0	0	1	1	0
0	1	0	1	1	1	1	0	0	1	1	1
0	1	1	0	0	1	1	1	0	0	1	1
0	1	1	0	1	0	1	1	0	0	1	0
0	1	1	0	1	1	1	1	0	0	1	1
0	1	1	1	0	0	0	1	0	0	1	0
0	1	1	1	0	1	0	1	0	0	1	0
0	1	1	1	1	0	1	1	0	0	1	1
1	0	0	0	0	0	1	0	0	0	1	0
1	0	0	0	0	1	1	0	0	0	1	1
1	0	0	0	1	0	1	0	0	0	1	0
1	0	0	0	1	1	1	0	0	0	1	1
1	0	0	1	0	0	1	0	0	1	1	0
1	0	0	1	0	1	1	0	0	1	1	1

1	0	0	1	1	0	1	0	0	1	1	0
1	0	0	1	1	1	0	0	1	1	1	
1	0	1	0	0	0	1	1	0	0	1	0
1	0	1	0	0	1	1	1	0	0	1	1
1	0	1	0	1	0	1	1	0	0	1	0
1	0	1	0	1	1	1	1	0	0	1	1
1	0	1	1	0	0	1	1	0	1	1	0
1	0	1	1	0	1	1	1	0	1	1	1
1	0	1	1	1	0	1	1	0	1	1	0
1	0	1	1	1	1	1	1	0	1	1	1
1	1	0	0	0	0	1	1	0	0	1	0
1	1	0	0	0	1	1	1	0	0	1	1
1	1	0	0	1	0	1	1	1	0	1	0
1	1	0	0	1	1	1	1	1			

## Step 6 Logic Diagram



u3312671-Bazla  
Bilquees-Assignment

## Step 7 Python Code

```
# Name: Bazla Bilquees
# Student ID: u3312671
# Case Study: Campus Café Checkout

# Prices menu: item -> (price, category)
menu = {
    "Coffee": (3.50, "Drink"),
    "Tea": (2.50, "Drink"),
    "Muffin": (2.00, "Food"),
    "Sandwich": (5.00, "Food"),
    "Smoothie": (4.00, "Drink"),
    "Salad": (4.50, "Food")
}

cart = []

# Function to display the menu
def show_menu():
    print("\n--- Café Menu ---")
    for item, (price, category) in menu.items():
        print(f'{item:10} - ${price:.2f} ({category})')
    print("-----")
```

```

# Add item to cart
def add_item(cart):
    show_menu()
    item = input("Enter item to add: ").title()
    if item in menu:
        qty = input("Quantity (default 1): ").strip()
        qty = int(qty) if qty.isdigit() and int(qty) > 0 else 1
        cart.append((item, qty))
        print(f'{qty} x {item} added to cart.')
    else:
        print("Item not on the menu.")

# View current cart
def view_cart(cart):
    if not cart:
        print("Cart is empty.")
        return
    print("\n--- Current Cart ---")
    for item, qty in cart:
        price = menu[item][0]
        print(f'{qty} x {item} - ${price*qty:.2f}')
        print("-----")

# Checkout and print receipt
def checkout(cart):
    if not cart:
        print("Cart is empty. Nothing to checkout.")
        return

    subtotal = sum(menu[item][0]*qty for item, qty in cart)
    tax = subtotal * 0.10
    discount = 0.0

    # Student discount
    student = input("Apply student discount (5%)? (y/n): ").lower()
    if student == "y":
        discount = (subtotal + tax) * 0.05

    # Meal deal discount: $2 off per drink+food combo
    drinks = sum(qty for item, qty in cart if menu[item][1] == "Drink")
    foods = sum(qty for item, qty in cart if menu[item][1] == "Food")
    meal_deal_discount = 0
    combos = min(drinks, foods)
    if combos >= 1:
        meal_deal_discount = combos * 2.00

```

```

total = subtotal + tax - discount - meal_deal_discount

# Print receipt
print("\n--- Receipt ---")
for item, qty in cart:
    price = menu[item][0]
    print(f'{qty} x {item} - ${price*qty:.2f}')
print(f"Subtotal: ${subtotal:.2f}")
print(f"Tax (10%): ${tax:.2f}")
if discount > 0:
    print(f"Student discount: -$ {discount:.2f}")
if meal_deal_discount > 0:
    print(f"Meal deal discount: -$ {meal_deal_discount:.2f}")
print(f"Total: ${total:.2f}")
print("Thank you for visiting!")

# Main program loop
def main():
    while True:
        print("\n--- Café POS ---")
        print("1. Show Menu\n2. Add Item\n3. View Cart\n4. Checkout\n5. Exit")
        choice = input("Enter choice: ").strip()
        if choice == "1":
            show_menu()
        elif choice == "2":
            add_item(cart)
        elif choice == "3":
            view_cart(cart)
        elif choice == "4":
            checkout(cart)
            cart.clear()
        elif choice == "5":
            print("Exiting program.")
            break
        else:
            print("Invalid choice. Try again.")

if __name__ == "__main__":
    main()

```

**Step 8 -Testing:** handwritten expected results + test runs & notes

## Case Study -1 Assignment 2.

Run:

STEP #06.

1. Show menu:
2. Add Item
3. View Cart
4. Checkout
5. Exit.

After choosing option 1 it will show the menu with price.

Coffee — \$3.50  
 Tea — \$2.50  
 Muffin — \$2.00  
 Sandwich — \$5.00  
 Smoothie — \$4.00  
 Salad — \$4.50

After that when we choose option 2 to add item into the cart it will ask the quantity of that item for example if i choose tea it will show me like this:

Enter item to add: Tea  
 Quantity = 4

4 x Tea added to cart  
 If i want to add another item i have to repeat the above procedure after that when i choose option 3 it will show the display:

$$4 \times \text{Tea} = \$10.00$$

$$4 \times \text{Muffin} = \$8.00$$

After choosing option 4 it will ask about the student discount (5%)

$$4 \times \text{Tea} = \$10.00$$

$$4 \times \text{Muffin} = \$8.00$$

$$\text{Subtotal} = \$18.00$$

$$\text{Tax (10\%)} = \$1.80$$

$$\text{Student discount (5\%)} = \$-0.99$$

$$\text{Meal deal discount} = \$8.00$$

$$\text{Total} = \$10.81$$

? Meal deal discount  
 When food and drink both present into the cart.

```
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Python 3.13.7 (tags/v3.13.7-bceelc3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.

>>> = RESTART: C:/Users/Lenovo/Desktop/u3312671-Bazla Bilquees- Assignment 2- Case study 1.py

--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 1

--- Café Menu ---
Coffee - $3.50 (Drink)
Tea - $2.50 (Drink)
Muffin - $2.00 (Food)
Sandwich - $5.00 (Food)
Smoothie - $4.00 (Drink)
Salad - $4.50 (Food)
-----
--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 2

--- Café Menu ---
Coffee - $3.50 (Drink)
Tea - $2.50 (Drink)
Muffin - $2.00 (Food)
Sandwich - $5.00 (Food)
Smoothie - $4.00 (Drink)
Salad - $4.50 (Food)
-----
Enter item to add: Tea
Quantity (default 1): 4
4 x Tea added to cart.
```

Activate Windows  
Go to Settings to activate Windows.

Ln: 103 Col: 0

```
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Smoothie - $4.00 (Drink)
Salad - $4.50 (Food)
-----
Enter item to add: Tea
Quantity (default 1): 4
4 x Tea added to cart.

--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 2

--- Café Menu ---
Coffee - $3.50 (Drink)
Tea - $2.50 (Drink)
Muffin - $2.00 (Food)
Sandwich - $5.00 (Food)
Smoothie - $4.00 (Drink)
Salad - $4.50 (Food)
-----
Enter item to add: Muffin
Quantity (default 1): 4
4 x Muffin added to cart.

--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 3

--- Current Cart ---
4 x Tea - $10.00
4 x Muffin - $8.00
-----
--- Café POS ---
```

Activate Windows  
Go to Settings to activate Windows.

Ln: 103 Col: 0

```

IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 3

--- Current Cart ---
4 x Tea - $10.00
4 x Muffin - $8.00
-----
--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 4
Apply student discount (5%)? (y/n): y

--- Receipt ---
4 x Tea - $10.00
4 x Muffin - $8.00
Subtotal: $18.00
Tax (10%): $1.80
Student discount: -$0.99
Meal deal discount: -$8.00
Total: $10.01
Thank you for visiting!

--- Café POS ---
1. Show Menu
2. Add Item
3. View Cart
4. Checkout
5. Exit
Enter choice: 5
Exiting program.

```

Activate Windows  
Go to Settings to activate Windows.

Ln: 103 Col: 0

5:52 PM 9/13/2025

## Step 9- Refinement via GenAI: prompt(s), what changed, justification

The Python Cafe checkout system has been improved with a number of practical features that greatly improve user experience and logic of operation when compared with the initial system. The system used to be limited to calculating a simple subtotal and providing a fixed tax rate as well, which did not offer flexibility and incentives to the customers. The new version has introduced a meal-deal discount and automatically a 10 percent discount is offered in case both food and beverage products are in the cart. This will stimulate the selling of multiple products and capture real world promotion. The other significant upgrade is the optional student discount, whereby users can make sure that they are entitled to an extra 5 percent off. This substitutes the inflexible nature of the old system, which used a discounts calculation without the user involvement. In addition, any money values are now in a uniform format of two decimal places as:.2f, which makes receipt readability and professionalism more agreeable. There is also the introduction of a loyalty points program in the new version where one point is earned after every five dollars are spent. This enhances the check out process and retention of customers. Finally, users have the option of saving their receipt which mimics the real world digital receipt storage. Compared to the original, the new system is more dynamic, easier to use and commercially feasible- it turns a mere calculator into a smart, customer-oriented checkout helper.

This document outlines the improvements made to a basic Python café checkout system. The original version included simple subtotal and tax calculations. The enhanced version introduces:

- Meal-deal discount logic
- Optional student discount
- Consistent currency formatting
- Safer input validation
- Loyalty points system

Message Copilot

Smart (GPT-5) ▾

+ 0 Activate Windows  
Go to Settings to activate Windows.

7:31 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jyqj

Feature Before After

Message Copilot

Smart (GPT-5) ▾

+ 0 Activate Windows  
Go to Settings to activate Windows.

7:32 AM 9/11/2025

Message Copilot

Smart (GPT-5) + Activate Windows Go to Settings to activate Windows.

Feature	Before	After
Currency Formatting	Inconsistent	All monetary values formatted to two decimals using <code>:.2f</code>
Meal-Deal Discount	Not present	10% off if cart contains both food and drink
Student Discount	Always prompted	Now optional and applied only if user confirms
Input Validation	Basic	Improved: safer quantity input and choice handling
Loyalty Points	Not available	Earn 1 point per \$5 spent
Receipt Saving	Not available	↓ Option to save receipt after checkout

## Code Structure

### 1. Menu Definition

```
Python ^ Copy
menu = {
    "Coffee": (3.50, "Drink"),
    "Tea": (2.75, "Drink"),
    "Sandwich": (5.00, "Food"),
    "Salad": (4.50, "Food"),
    "Cookie": (1.25, "Food"),
    "Juice": (3.00, "Drink")
}
```

Message Copilot

Smart (GPT-5) + Activate Windows Go to Settings to activate Windows.

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryq

Message Copilot

Smart (GPT-5) + Activate Windows Go to Settings to activate Windows.

Feature	Before	After
Loyalty Points	Not available	Earn 1 point per \$5 spent
Receipt Saving	Not available	Option to save receipt after checkout

1. Menu Definition

```
Python ^ Copy
menu = {
    "Coffee": (3.50, "Drink"),
    "Tea": (2.75, "Drink"),
    "Sandwich": (5.00, "Food"),
    "Salad": (4.50, "Food"),
    "Cookie": (1.25, "Food"),
    "Juice": (3.00, "Drink")
}
```

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

2. Meal-Deal Detection

```
Python ⏪ Copy
```

```
def has_meal_deal(cart):
    categories = set(menu[item][1] for item, qty in cart)
    return "Food" in categories and "Drink" in categories
```

3. Cart Builder

```
Python ⏪ Copy
```

```
def build_cart():
```

Message Copilot

Smart (GPT-5) + 0 Activate Windows Go to Settings to activate Windows.

7:34 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

return "Food" in categories and "Drink" in categories

3. Cart Builder

```
Python ⏪ Copy
```

```
def build_cart():
    cart = []
    print("Welcome to the Café! Here's our menu:")
    for item, (price, _) in menu.items():
        print(f"- {item}: ${price:.2f}")

    while True:
        choice = input("\nEnter item name to add (or 'done' to finish): ")
        if choice == "Done":
            break
        if choice not in menu:
```

Message Copilot

Smart (GPT-5) + 0 Activate Windows Go to Settings to activate Windows.

7:34 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

```
choice = input("\nEnter item name to add (or 'done' to finish): ")
if choice == "Done":
    break
if choice not in menu:
    print("Item not found. Please choose from the menu.")
    continue
try:
    qty = int(input(f"Enter quantity for {choice}: "))
    if qty <= 0:
        raise ValueError
except ValueError:
    print("Invalid quantity. Defaulting to 1.")
    qty = 1
cart.append((choice, qty))
return cart
```

Sign in

7:34 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

#### 4. Loyalty Points Calculation

```
Python ⌂ Copy
def calculate_loyalty_points(total):
    return int(total // 5)
```

#### 5. Receipt Generator

```
Python ⌂ Copy
def generate_receipt(cart, subtotal, tax, discount, total):
    lines = ["--- ┌ Receipt ---"]
    for item, qty in cart:
```

Message Copilot

Smart (GPT-5) + 0 Activate Windows Go to Settings to activate Windows.

7:35 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

#### 5. Receipt Generator

```
Python ⌂ Copy
def generate_receipt(cart, subtotal, tax, discount, total):
    lines = ["--- ┌ Receipt ---"]
    for item, qty in cart:
        price = menu[item][0]
        lines.append(f'{qty} x {item} - ${price * qty:.2f}')
    lines.append(f"Subtotal: ${subtotal:.2f}")
    lines.append(f"Tax (10%): ${tax:.2f}")
    lines.append(f"Total Discounts: -${discount:.2f}")
    lines.append(f"Total: ${total:.2f}")
    return "\n".join(lines)
```

#### 6. Checkout Function

Message Copilot

Smart (GPT-5) + 0 Activate Windows Go to Settings to activate Windows.

7:35 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

## 6. Checkout Function

```
Python ^ ⚡ Copy
```

```
def checkout(cart):
    if not cart:
        print("Cart is empty. Nothing to checkout.")
        return

    subtotal = sum(menu[item][0] * qty for item, qty in cart)
    tax = subtotal * 0.10
    discount = 0.0

    if has_meal_deal(cart):
        meal_deal_discount = subtotal * 0.10
        discount += meal_deal_discount
        print("Meal-deal discount applied: -${:.2f}".format(meal_deal_discount))
```

Message Copilot

Smart (GPT-5) + 0 Activate Windows Go to Settings to activate Windows.

7:35 AM 9/11/2025

copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj

```
meal_deal_discount = subtotal * 0.10
discount += meal_deal_discount
print("Meal-deal discount applied: -${:.2f}".format(meal_deal_discount))

student = input("Apply student discount (5%)? (y/n): ").strip().lower()
if student == "y":
    student_discount = subtotal * 0.05
    discount += student_discount
    print("Student discount applied: -${:.2f}".format(student_discount))

total = subtotal + tax - discount
points = calculate_loyalty_points(total)
receipt = generate_receipt(cart, subtotal, tax, discount, total)

print("\n" + receipt)
print("★ Loyalty Points Earned: {}".format(points))
print("Thank you for visiting!")
```

Message Copilot

Smart (GPT-5) + 0 Activate Windows Go to Settings to activate Windows.

7:35 AM 9/11/2025

A screenshot of the Microsoft Copilot interface. At the top, the URL is copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj. The main area displays a Python script for a loyalty program:

```
print(f"★ Loyalty Points Earned: {points}")
print("Thank you for visiting!")

save = input("Would you like to save your receipt? (y/n): ").strip().lower()
if save == "y":
    print("\nReceipt saved:\n")
    print(receipt)
```

## 7. Main Program

```
Python ^ Copy

if __name__ == "__main__":
    cart = build_cart()
    checkout(cart)
```

Message Copilot

Smart (GPT-5) ▾

+ 0 Activate Windows  
Go to Settings to activate Windows.



## ✓ Sample Interaction

A screenshot of the Microsoft Copilot interface. The main area shows a sample interaction session:

```
Enter item name to add (or 'done' to finish): Coffee
Enter quantity for Coffee: 2
Enter item name to add (or 'done' to finish): Sandwich
Enter quantity for Sandwich: 1
Enter item name to add (or 'done' to finish): done
Apply student discount (5%)? (y/n): y
Would you like to save your receipt? (y/n): y
```

Output:

A screenshot of the Microsoft Copilot interface. The main area shows a sample interaction session:

```
Code ^ Copy
```

Message Copilot

Smart (GPT-5) ▾

+ 0 Activate Windows  
Go to Settings to activate Windows.



The screenshot shows a Microsoft Copilot interface. At the top, there's a navigation bar with back, forward, and search icons, followed by the URL 'copilot.microsoft.com/chats/JghnA8q1LPPcs2k8Jryqj'. To the right are icons for star, copy, download, and more. Below the URL is a sign-in button. The main area has a title 'Output:' and a 'Code' dropdown. The code output is a receipt with the following details:

```

    Meal-deal discount applied: -$1.05
    Student discount applied: -$0.52

    --- ■ Receipt ---
    2 x Coffee - $7.00
    1 x Sandwich - $5.00
    Subtotal: $12.00
    Tax (10%): $1.20
    Total Discounts: -$1.57
    Total: $11.63
    ★ Loyalty Points Earned: 2
    Thank you for visiting!
    ■ Receipt saved:

```

Below the code output is a message input field labeled 'Message Copilot' with a 'Smart (GPT-5)' dropdown. To the right, there's an 'Activate Windows' button with a link to 'Go to Settings to activate Windows.' The bottom of the screen shows a taskbar with various pinned apps like File Explorer, Edge, and Mail, along with system icons for battery, signal, and volume.

Bazla Bilquees

U3312671

Introduction To Information Technology

## **ASSIGNMENT # 02 – PYTHON FUNDAMENTAL**

### **Case Study 2 – Smart Classroom Monitor**

#### **Step 1 — Understand the Problem:**

Create a console program to monitor classroom state: projector status, topic, attendance management, temperature log and alert for over-capacity and out-of-range temperature. Produce a report summarizing the room state.

#### **Step 2 — Inputs & Outputs**

Variable	Device	Type	Unit	Logic	Description
Projector statue	Projector switch	Input	Boolean	On=1, OFF=0	Give the status of the project 1= On, 0=Off
Capacity	System Database	Input	Students/Persons	>or equal to 0	Maximum seating capacity of classroom
Attendance Names	RFID/Manual Entry	Input	Students	-	Students attending class
Topic	Instructor Input	Input	-	-	Lecture topic name

Temperature Log	Temperature Sensor	Input	C	Numeric	Records classroom temperature samples
Attendance Count	System (processor)	Output	Students	>or equal to 0	Number of students in the class
Capacity Alert	LCD/ Screen/ Alarm	Output	Boolean	Full=1, Normal=0	Warning if attendance exceeds capacity.
Temp Statics	System (processor)	Output	C	Min/Max/Avg	Temperature summary of classroom
Temp Alert	Alarm/SMS	Output	Boolean	Low=1, High=1, Normal=0	Alert if temperature <16C or >28C
Projector Reminder	Display	Output	Text	Reminder=1	If projector OFF while topic assigned

## Step 3 — Algorithm

Initialized system state

- Set projector\_on = False
- Set topic = None
- Set attendance = {} (empty set)
- Preferably, set temperature\_log = [] (list).
- Maximum number of students allowed.

Main menu and menu options:

- Toggle projector
- Set topic
- Add student
- Remove student
- Add temperature reading
- Show report
- Exit

Continue till user selects Exit:

- If Toggle projector:
- Change projector on/off ON/Off.

- If Set topic:
- Enter a string value and save in topic.

If Add student:

- Input student name.
- Add name to attendance set.

If Remove student:

- Input student name.
- Check-out in case of attendance.

If Add temperature reading:

- Input float value ( $^{\circ}\text{C}$ ).
- Append to temperature\_log.

If Show report:

- Record the number of students in attendance.
- count bigger than capacity: print "ROOM FULL".
- Computations of temperature statistics = (min,max,average).
- When any temperature of less than 16 C or more than 28 C occurs, then display temperature caution.
- When setting topic and projector off then display reminder Projector is OFF topic is set.

If Exit chosen:

- Stop the program.
- End.

## Step 4 — Flow Chart



## Step 4 – PSEUDOCODE

```

BEGIN
ROOM ← {'projector_on': False, 'capacity': 30, 'topic': ""}
attendance ← set()
temperatures ← []
WHILE True
print menu
GET choice
IF toggle projector → toggle_projector()
IF set topic → set_topic(input)
IF add student → add_student(name) [check capacity]
IF remove student → remove_student(name)

```

```

IF add temp → add_temp(value) [warn if out-of-range]
IF report → report()
IF exit → break
END WHILE
END

```

## Step 5 Python Code

```

#Name: Bazla Bilquees
#Student ID: u3312671
#Case Study 2: Smart Classroom Monitor (robust implementation)
import sys
import traceback
from datetime import datetime
# -Room state and data structures -
ROOM = {
    "projector_on": False, # bool
    "capacity": 30,        # int
    "topic": ""           # str
}

attendance = set()      # set of student names (strings)
temperatures = []        # list of float temperature readings (°C)

# ----- Input helpers -----
def safe_input(prompt):
    """
    Wrapper for input() which handles Ctrl+C / Ctrl+D gracefully.
    Returns a stripped string, or None if the user cancelled.
    """
    try:
        s = input(prompt)
    except (EOFError, KeyboardInterrupt):
        print("\nInput cancelled by user. Exiting program.")
        return None
    return s.strip()

def get_nonempty_string(prompt):
    while True:
        s = safe_input(prompt)
        if s is None:
            return None
        if s != "":
            return s
        print("Please enter a non-empty value.")

# - Classroom operations -
def toggle_projector():

```

```

ROOM["projector_on"] = not ROOM["projector_on"]
state = "ON" if ROOM["projector_on"] else "OFF"
print(f"Projector switched {state}.")
```

```

def set_topic():
    topic = get_nonempty_string("Enter lecture topic (or blank to cancel): ")
    if topic is None:
        return
    ROOM["topic"] = topic
    print(f"Topic set to: {ROOM['topic']}")
```

```

def add_student():
    name = get_nonempty_string("Enter student name to add: ")
    if name is None:
        return
    if len(attendance) >= ROOM["capacity"]:
        print("⚠ ROOM FULL — cannot add more students.")
        return
    if name in attendance:
        print(f"{name} is already marked present (no duplicate entries).")
    else:
        attendance.add(name)
        print(f"Added {name}. Attendance count: {len(attendance)}/{ROOM['capacity']}")
```

```

def remove_student():
    name = get_nonempty_string("Enter student name to remove: ")
    if name is None:
        return
    if name in attendance:
        attendance.remove(name)
        print(f"Removed {name}. Attendance count: {len(attendance)}/{ROOM['capacity']}")
```

```

    else:
        print(f"{name} not found in attendance.")
```

```

def add_temperature():
    s = safe_input("Enter temperature reading in °C (e.g. 22.5): ")
    if s is None:
        return
    try:
        t = float(s)
    except ValueError:
        print("Invalid number. Please enter a numeric temperature (e.g. 21.3).")
        return
    temperatures.append(t)
    print(f"Temperature {t:.1f}°C added. Total readings: {len(temperatures)}")
    if t < 16 or t > 28:
        print("⚠ Temperature out of recommended range (<16°C or >28°C).")
```

```

def temp_stats():
```

```

if not temperatures:
    return None, None, None
tmin = min(temperatures)
tmax = max(temperatures)
tavg = sum(temperatures) / len(temperatures)
return tmin, tmax, tavg
def show_temperature_stats():
    tmin, tmax, tavg = temp_stats()
    if tmin is None:
        print("No temperature readings available.")
        return
    print(f"Temperature stats — Min: {tmin:.2f}°C, Max: {tmax:.2f}°C, Avg: {tavg:.2f}°C")
    if tmin < 16 or tmax > 28:
        print("⚠ ALERT: Temperature readings out of recommended range.")
def report():
    print("\n" + "=" * 36)
    print("CLASSROOM REPORT")
    print(f"Projector: {'ON' if ROOM['projector_on'] else 'OFF'}")
    print(f"Topic: {ROOM['topic']} or '<none>'")
    print(f"Capacity: {ROOM['capacity']} ")
    print(f"Attendance ({len(attendance)}): {', '.join(sorted(attendance))} or '<none>'")
    tmin, tmax, tavg = temp_stats()
    if tmin is None:
        print("Temperature readings: <none>")
    else:
        print(f"Temperature: min={tmin:.2f}°C, max={tmax:.2f}°C, avg={tavg:.2f}°C")
        if tmin < 16 or tmax > 28:
            print("⚠ Temperature ALERT: values out of range (<16°C or >28°C).")
    # Alerts
    if len(attendance) > ROOM["capacity"]:
        print("⚠ ROOM OVER CAPACITY!")
    if ROOM["topic"] and not ROOM["projector_on"]:
        print("⚠ Reminder: Topic is set but projector is OFF.")
    print("=" * 36 + "\n")

# -Utility: change capacity (optional) -
def set_capacity():
    s = safe_input(f"Enter new capacity (current {ROOM['capacity']}) or blank to cancel: ")
    if s is None or s == "":
        return
    try:
        val = int(s)
        if val < 0:
            print("Capacity must be a non-negative integer.")
        return
    
```

```

except ValueError:
    print("Please enter an integer value.")
    return
ROOM['capacity'] = val
print(f"Capacity set to {ROOM['capacity']}")

# - Main program loop -
def main():
    MENU_TEXT = """\nSMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit
"""

    while True:
        print(MENU_TEXT)
        choice = safe_input("Choose option (1-9): ")
        if choice is None:
            # user cancelled input (Ctrl+C / Ctrl+D)
            break
        choice = choice.strip()
        if choice == "1":
            toggle_projector()
        elif choice == "2":
            set_topic()
        elif choice == "3":
            add_student()
        elif choice == "4":
            remove_student()
        elif choice == "5":
            add_temperature()
        elif choice == "6":
            show_temperature_stats()
        elif choice == "7":
            report()
        elif choice == "8":
            set_capacity()
        elif choice == "9":
            print("Exiting Smart Classroom Monitor. Goodbye!")
            break
        else:
            # allow text commands too

```

```

cmd = choice.lower()
if cmd in ("toggle", "projector"):
    toggle_projector()
elif cmd == "exit":
    break
else:
    print("Invalid choice — enter 1-9 or a command (toggle/exit).")
# -Program entry with top-level exception handling -
if __name__ == "__main__":
    try:
        main()
    except Exception as e:
        # print friendly error message and record full traceback to a log file
        print("An unexpected error occurred. A log file 'classroom_error.log' was created.")
        with open("classroom_error.log", "a", encoding="utf-8") as f:
            f.write(f"\n[ {datetime.now().isoformat()} ] Unhandled exception:\n")
            traceback.print_exc(file=f)
        # also print the traceback snippet to console for quick debugging
        traceback.print_exc()
        sys.exit(1)

```

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



u3312671-Bazla  
Bilquees-Case study 2

U3312671 - Barza Bilquees

IIT

## Case Study-II - Assignment 2

STEP#06 Test No.	Action / Input	Expected Result	Actual Result	Status.
1.	Start Program	Menu will displayed with 0-9 options	Displayed Correctly	Pass.
2.	Projector Status	1 <sup>st</sup> press → ON, 2 <sup>nd</sup> press → OFF	Messages correct	Pass.
3.	Enter Topic "Animals"	Output shows "Topic set to Animals".	Matches	Pass
4.	Add student name "Ayesha"	student list shows "Ayesha" added	Matches	Pass.
5.	Remove the name of student "Ayesha"	Name Removed	Matches	Pass.
6.	Add temperature reading '26'	Value stored	Matches	Pass.
7.	Show statistic	Displays topic, projector status, total student, avg. temp	Matches	Pass.
8.	Generate Complete Report	Formatted report	Matches	Pass.
9.	Exit	Program terminates without error	Matches	Pass.

```
idle Shell 3.13.7
File Edit Shell Debug Options Window Help
Python 3.13.7 (tags/v3.13.7-bceelc3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.

>>> ===== RESTART: C:/Users/Lenovo/Desktop/Case Study 2.Assignment 2py.py =====

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 1
Projector switched ON.

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 2
Enter lecture topic (or blank to cancel): Animals
Topic set to: Animals

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Ln: 168 Col: 0

idle Shell 3.13.7
File Edit Shell Debug Options Window Help
Choose option (1-9): 3
Enter student name to add: Ayesha
Added Ayesha. Attendance count: 1/30

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 4
Enter student name to remove: Ayesha
Removed Ayesha. Attendance count: 0/30

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 5
Enter temperature reading in °C (e.g. 22.5): 25
Temperature 25.0°C added. Total readings: 1

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Ln: 168 Col: 0

1:45 PM 9/11/2025
```

```
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 6
Temperature stats - Min: 25.00°C, Max: 25.00°C, Avg: 25.00°C

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 7

=====
CLASSROOM REPORT
Projector: ON
Topic: Animals
Capacity: 30
Attendance (0): <none>
Temperature: min=25.00°C, max=25.00°C, avg=25.00°C
=====

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Ln: 168 Col: 0

```

```
IDLE Shell 3.13.7
File Edit Shell Debug Options Window Help
Choose option (1-9): 8
Enter new capacity (current 30) or blank to cancel: 35
Capacity set to 35

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 7

=====
CLASSROOM REPORT
Projector: ON
Topic: Animals
Capacity: 35
Attendance (0): <none>
Temperature: min=25.00°C, max=25.00°C, avg=25.00°C
=====

SMART CLASSROOM MONITOR
1) Toggle projector
2) Set topic
3) Add student
4) Remove student
5) Add temperature reading
6) Show temperature stats
7) Show full report
8) Set capacity (optional)
9) Exit

Choose option (1-9): 9
Exiting Smart Classroom Monitor. Goodbye!
>>>

Ln: 168 Col: 0

```

## Step 7- Truth Table & Boolean Expression:

The classroom is “ready to generate a valid report” only when:

- Projector is ON (P)

- Topic is set (T)
- At least one student is present (S)
- At least one temperature reading exists (R)

Therefore:

The Boolean expression is:

$M = P \cdot T \cdot S \cdot R$  (Where “.” means logical AND)

P	T	S	R	MON
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

## Step 8 – LOGIC DIAGRAM

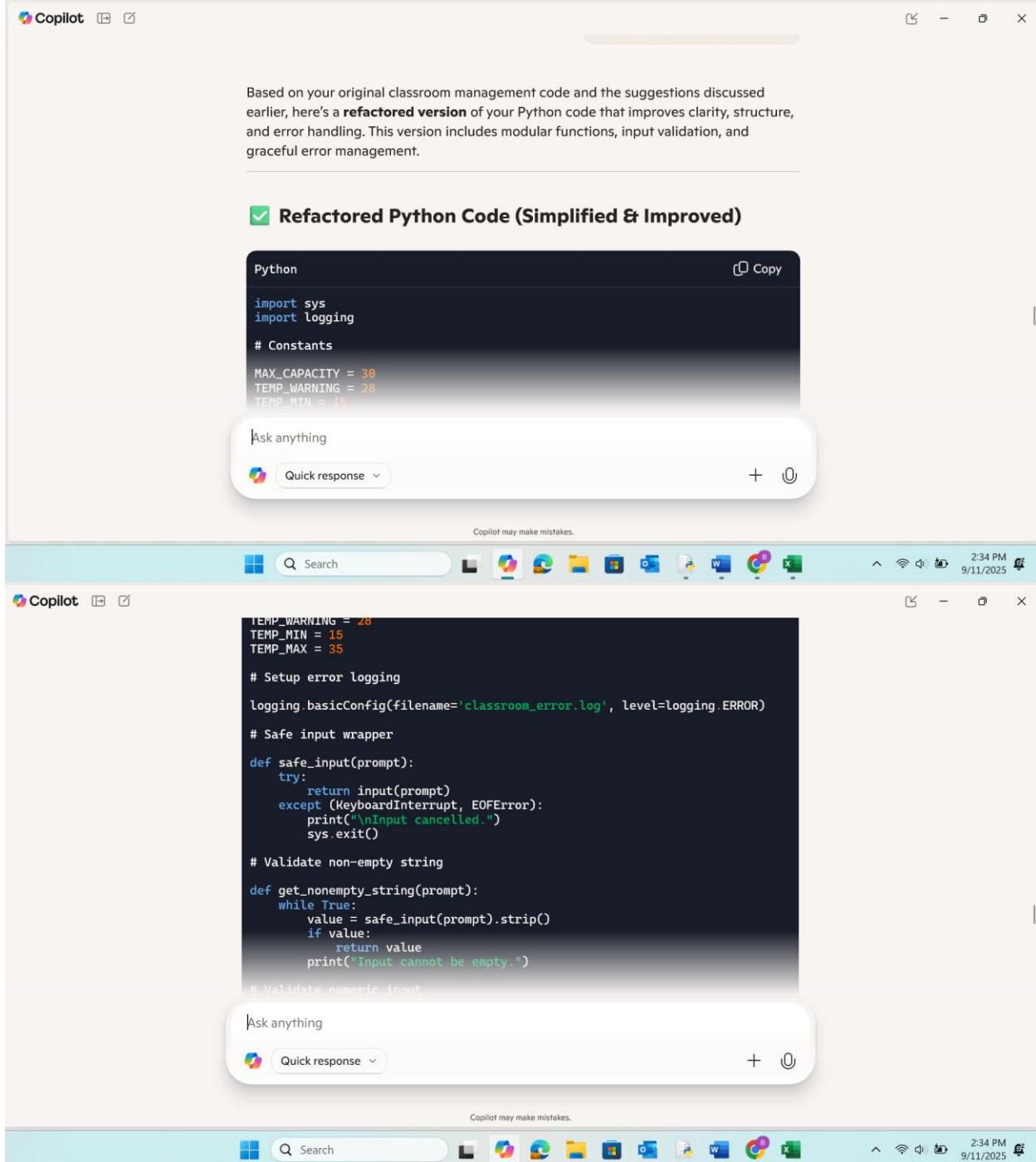


U3312671- Bazla  
Bilquees- LOGIC DIAG

## Step 9 - Refinement via GenAI: prompt(s), what changed, justification

GenAI helped me to make my classroom management code clearer, structured, and strong, to refine it. I requested assistance with the logic modularisation, improved input validation, and graceful behaviour of the user interruptions. Consequently, I rearranged the code into clear functions like adding students, switching projectors and creating report to enhance readability and maintainability. To avoid unexpected exits (such as Ctrl+C or Ctrl+D), I used safe input() and I implemented checks to avoid empty or invalid entries, particularly student names and numeric inputs such as temperature and capacity. Threshold values were made constant so that crashes later may be more easily updated, and a top-level exception handler was introduced to record errors without terminating the program. These are refinements that make user interaction smoother, the data more resilient and the standard of the code more professional. The advice of

GenAI has enabled me to develop a practical code into a more formal, trustworthy system that is easier to debug, expand and present with confidence in real-life scenario.



The image shows three identical screenshots of the Microsoft Copilot interface, each displaying a refactored Python script. The top part of each screenshot contains a message from Copilot stating: "Based on your original classroom management code and the suggestions discussed earlier, here's a **refactored version** of your Python code that improves clarity, structure, and error handling. This version includes modular functions, input validation, and graceful error management." Below this message is a section titled "Refactored Python Code (Simplified & Improved)" with a checkmark icon. The code itself is as follows:

```
Python
import sys
import logging

# Constants
MAX_CAPACITY = 30
TEMP_WARNING = 28
TEMP_MIN = 15

# Setup error logging
logging.basicConfig(filename='classroom_error.log', level=logging.ERROR)

# Safe input wrapper
def safe_input(prompt):
    try:
        return input(prompt)
    except (KeyboardInterrupt, EOFError):
        print("\nInput cancelled.")
        sys.exit()

# Validate non-empty string
def get_nonempty_string(prompt):
    while True:
        value = safe_input(prompt).strip()
        if value:
            return value
        print("Input cannot be empty.")

# Validate numeric input
def get_numeric_input(prompt):
    while True:
        value = safe_input(prompt).strip()
        if value:
            try:
                return float(value)
            except ValueError:
                print("Please enter a valid number.")


Ask anything
Quick response + 0
```

Copilot may make mistakes.

2:34 PM 9/11/2025

```
# Validate numeric input

def get_valid_number(prompt, min_val=None, max_val=None):
    while True:
        try:
            value = float(safe_input(prompt))
            if (min_val is not None and value < min_val) or (max_val is not None and value > max_val):
                print(f"Value must be between {min_val} and {max_val}.")
            else:
                return value
        except ValueError:
            print("Please enter a valid number.")

# Toggle projector

def toggle_projector(state):
    return not state

# Add student

def add_student(student_list):
    name = get_nonempty_string("Enter student name: ")
    student_list.append(name)
    print(f"{name} added successfully.")
```

Ask anything

Quick response ▾



```
print(f"Projector: {'On' if projector_on else 'Off'}")
print(f"Temperature: {temperature}°C")
print(f"Students Present: {len(students)}")
print("Status: " + ("Ready " + checked_mark if ready else "Not Ready " + crossed_out_x))

# Main loop

def main():
    projector_on = False
    students = []
    temperature = 25.0

    try:
        while True:
            print("\nMenu:")
            print("1. Toggle Projector")
            print("2. Set Temperature")
            print("3. Add Student")
            print("4. Generate Report")
            print("5. Exit")

            choice = safe_input("Choose an option (1-5): ")

            if choice == '1':
                projector_on = toggle_projector(projector_on)
                print(f"Projector is now {'On' if projector_on else 'OFF'}")
```

Ask anything

Quick response ▾



```
projector_on = toggle_projector(projector_on)
print("Projector is now {'On' if projector_on else 'Off'}")
elif choice == '2':
    temperature = get_valid_number("Enter room temperature: ", TEMP_WARNING)
    if temperature > TEMP_WARNING:
        print("⚠ Warning: Temperature is high!")
elif choice == '3':
    if len(students) < MAX_CAPACITY:
        add_student(students)
    else:
        print("⚠ Classroom is at full capacity!")
elif choice == '4':
    generate_report(projector_on, temperature, students)
elif choice == '5':
    print("Exiting program.")
    break
else:
    print("Invalid choice. Please select from 1 to 5.")
except Exception as e:
    logging.error("Unexpected error: %s", str(e))
    print("An unexpected error occurred. Please check the log file.")

if __name__ == "__main__":
    main()
```

Ask anything

Quick response ▾



```
except Exception as e:
    logging.error("Unexpected error: %s", str(e))
    print("An unexpected error occurred. Please check the log file.")

if __name__ == "__main__":
    main()
```

### Summary of Improvements

- Modular functions for clarity and reuse
- Input validation for strings and numbers
- Graceful handling of Ctrl+C / Ctrl+D
- Error logging for unexpected issues
- Constants for easy threshold updates
- Clear report logic based on readiness conditions

Let me know if you'd like this version tailored further to match your assignment.

Ask anything

Quick response ▾



## 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:

Activate Windows  
Go to Settings to activate Windows

F8	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	A	B	C	Y															
2	0	0	0	1															
3	0	0	1	1															
4	0	1	0	1															
5	0	1	1	1															
6	1	0	0	0															
7	1	0	1	0															
8	1	1	0	0															
9	1	1	1	1															
10																			
11																			
12																			
13																			
14																			
15																			
16																			
17																			
18																			
19																			
20																			

Ready Accessibility: Good to go

Search 5:02 PM 9/11/2025

### For Circuit B:

Activate Windows  
BigAir activated Internet access

F7	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	P	Q	R	Z															
2	0	0	0	0															
3	0	0	1	0															
4	0	1	0	0															
5	0	1	1	0															
6	1	0	0	0															
7	1	0	1	1															
8	1	1	0	0															
9	1	1	1	1															
10																			
11																			
12																			
13																			
14																			
15																			
16																			
17																			
18																			
19																			
20																			

Ready Accessibility: Good to go

Search 5:04 PM 9/11/2025

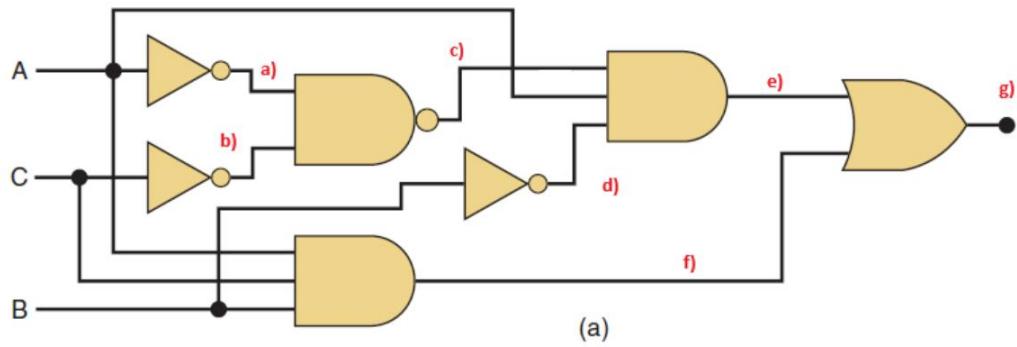
```

>>> 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=True
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
Activate Windows
Go to Settings to activate Windows.
Ln: 188 Col: 0
5:04 PM 9/11/2025

```

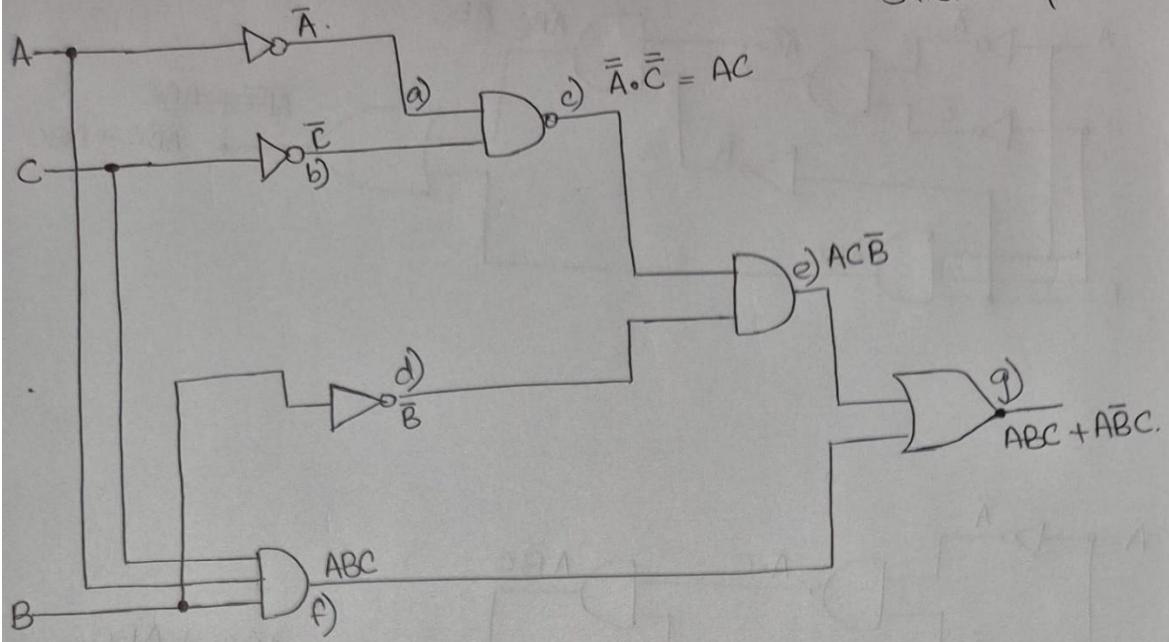
## Step 7- Boolean expression:

### For Circuit A:



U3312671

Bazla Bilquees



- a) A  $\bar{A}$
- b) C  $\bar{C}$
- c)  $\bar{A} \cdot \bar{C}$   $\bar{\bar{A}} \cdot \bar{\bar{C}} = AC$ .
- d) B  $\bar{B}$
- e)  $AC\bar{B}$   $A\bar{B}C$
- f) ABC  $A \cdot B \cdot C$
- g) e+f  $ABC + A\bar{B}\bar{C}$ .

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

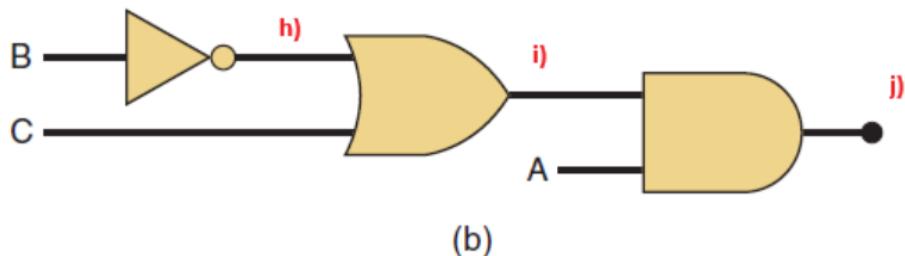
$$AC(B + \bar{B})$$

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

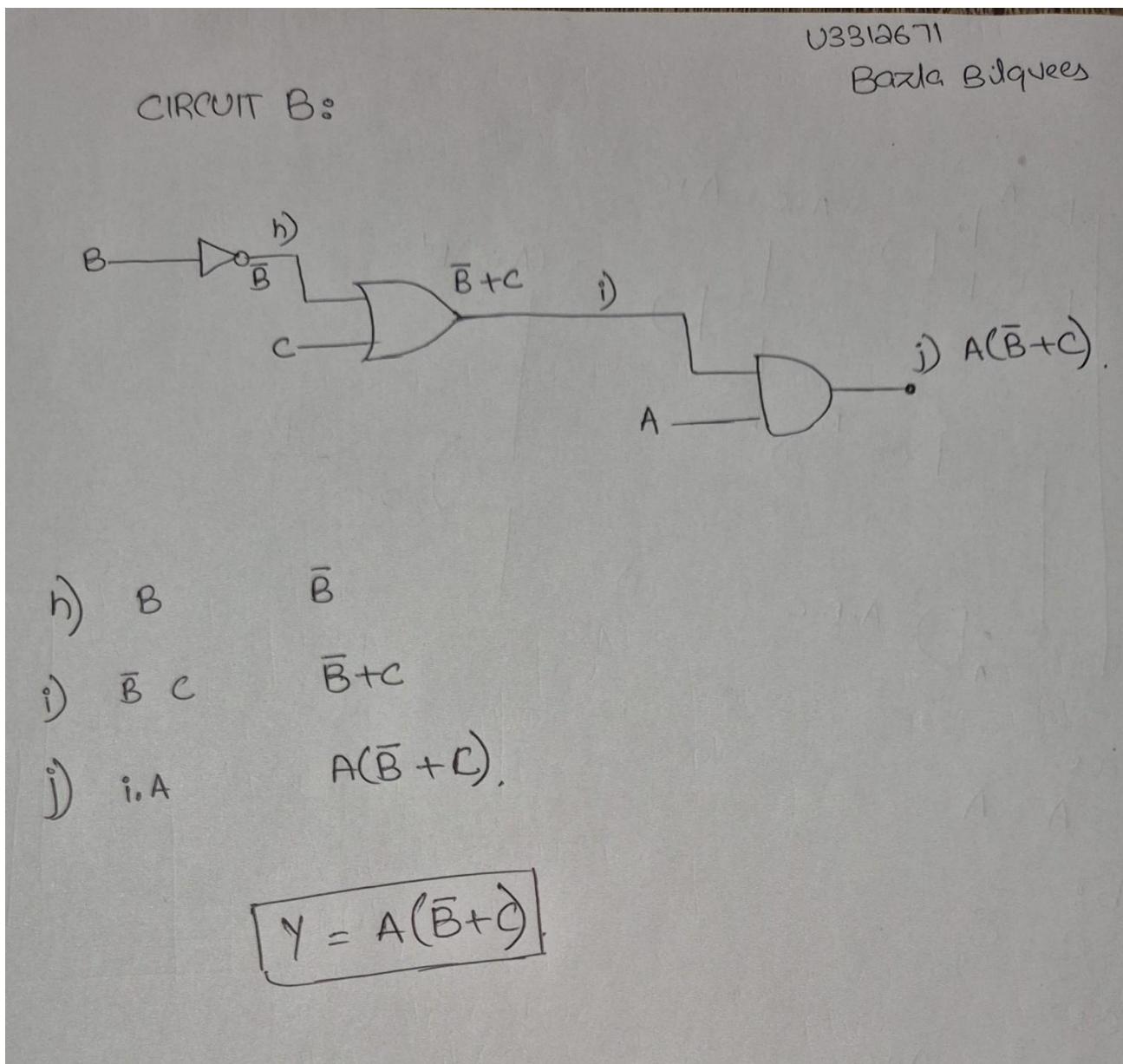
$$\boxed{* = AC}$$

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

```
>>> ===== 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=False
A=False, B=False, C=True => Y=True
A=False, B=False, C=False => Y=True
Activate Windows
Go to Settings to activate Windows.
Ln: 188 Col: 0
6:23 PM 9/11/2025
```

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

```
>>> ===== 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
Activate Windows
Go to Settings to activate Windows.
Ln: 188 Col: 0
6:24 PM 9/11/2025
```

## Step 9 - Check that both circuits are equivalent.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	A	B	C	X_A	Y_A	X_B	Y_B												
2	0	0	0	0	0	0	0												
3	0	0	1	1	0	1	0												
4	0	1	0	0	1	0	1												
5	0	1	1	1	1	1	1												
6	1	0	0	0	0	0	0												
7	1	0	1	1	0	1	0												
8	1	1	0	0	1	1	1												
9	1	1	1	1	1	1	1												
10																			
11																			
12																			
13																			
14																			
15																			
16																			
17																			
18																			
19																			
20																			

**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}")

```

IDLE Shell 3.13.7

File Edit Shell Debug Options Window Help

Python 3.13.7 (tags/v3.13.7:bce1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (AMD64)] on win32  
Enter "help" below or click "Help" above for more information.

>>> = RESTART: C:/Users/Lenovo/Desktop/u3312671-Bazla Bilquees- Assignment2 - Case Study 3 Step 4.py  
NOT equivalent. Differences found:  
Row 6: A=1 B=1 C=0 CircuitA=(0, 1) CircuitB=(1, 1)