|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Faculty of Information Technology** | | | | | | | | | |
| I declare that I am familiar with, and will abide to the Examination rules of CTU    **Signature** | **SUBJECT NAME: Programming with Python**  **SUBJECT CODE: PRP 411** | | | | | | | | |
| **Formative Assessment 1 Duration**:  **Date**:22-Jun-2023  **Total Marks**:  **Total pages**: | | | | **Examiner**:Mr.Junior Manganyi  **Moderator:** | | | | |
| **Student number** | | | | | | | | |
| 2 | 0 | 2 | 3 | 1 |  | 2 | 9 | 7 |
| **Surname**:Modise | | | | **Initials**: RGW | | | / | % |

Contents

[Code: 3](#_Toc138333813)

[Output: 8](#_Toc138333814)

# Code:

class Sale:

    def \_\_init\_\_(self):

        self.total\_sale = 0.0

class Table:

    def \_\_init\_\_(self, table\_num):

        self.table\_num = table\_num

        self.server = None

        self.customers = 0

        self.orders = []

    def assign\_server(self, server):

        self.server = server

    def add\_customers(self, count):

        self.customers = count

    def add\_order(self, order):

        self.orders.append(order)

    def prepare\_bill(self):

        total = sum(order.price \* order.quantity for order in self.orders)

        bill = f"Table: {self.table\_num}\n"

        bill += "----------------------\n"

        for order in self.orders:

            bill += f"{order.name} x {order.quantity}: R{order.price \* order.quantity}\n"

        bill += "----------------------\n"

        bill += f"Total: R{total}\n"

        return bill, total

    def clear\_table(self):

        self.server = None

        self.customers = 0

        self.orders = []

class Order:

    def \_\_init\_\_(self, name, price, quantity):

        self.name = name

        self.price = price

        self.quantity = quantity

def read\_login\_cred(file\_name):

    credentials = {}

    with open(file\_name, 'r') as file:

        for line in file:

            username, passwrd = line.strip().split(',')

            credentials[username] = passwrd

    return credentials

def read\_menu(file\_name):

    menu = {}

    with open(file\_name, 'r') as file:

        for line in file:

            item\_name, item\_price = line.strip().split(',')

            menu[item\_name] = float(item\_price)

    return menu

def display\_menu():

    print("Main Menu:")

    print("1. Assign Table")

    print("2. Change Customers")

    print("3. Add to Order")

    print("4. Prepare Bill")

    print("5. Complete Sale")

    print("6. Cash Up")

    print("0. Log Out")

def assign\_table(current\_server, table\_list):

    print("Available tables:")

    available\_tables = [table for table in table\_list if table.server is None]

    for table in available\_tables:

        print(f"Table {table.table\_num}")

    table\_num = int(input("Enter the table number: "))

    table = next((table for table in table\_list if table.table\_num == table\_num), None)

    if table and table.server is None:

        table.assign\_server(current\_server)

        choice = input("Do you want to add customers to the table? (y/n): ")

        if choice.lower() == "y":

            count = int(input("Enter the number of customers: "))

            table.add\_customers(count)

        print("Table assigned successfully!")

    else:

        print("Invalid table number or the table is already assigned.")

def change\_customers(current\_server, table\_list):

    print("Tables assigned to you:")

    assigned\_tables = [table for table in table\_list if table.server == current\_server]

    for table in assigned\_tables:

        print(f"Table {table.table\_num}")

    table\_num = int(input("Enter the table number: "))

    table = next((table for table in table\_list if table.table\_num == table\_num and table.server == current\_server), None)

    if table:

        count = int(input("Enter the new number of customers: "))

        table.add\_customers(count)

        print("Number of customers changed successfully!")

    else:

        print("Invalid table number or the table is not assigned to you.")

def add\_to\_order(current\_server, table\_list, menu\_items):

    print("Tables assigned to you:")

    assigned\_tables = [table for table in table\_list if table.server == current\_server]

    for table in assigned\_tables:

        print(f"Table {table.table\_num}")

    table\_num = int(input("Enter the table number: "))

    table = next((table for table in table\_list if table.table\_num == table\_num and table.server == current\_server), None)

    if table:

        print("Menu:")

        for item\_name, item\_price in menu\_items.items():

            print(f"{item\_name}: R{item\_price}")

        item\_name = input("Enter the name of the item: ")

        if item\_name in menu\_items:

            item\_price = menu\_items[item\_name]

            item\_quantity = int(input("Enter the quantity: "))

            order = Order(item\_name, item\_price, item\_quantity)

            table.add\_order(order)

            print("Order added successfully!")

        else:

            print("Invalid item name.")

    else:

        print("Invalid table number or the table is not assigned to you.")

def prepare\_bill(current\_server, table\_list):

    print("Tables assigned to you:")

    assigned\_tables = [table for table in table\_list if table.server == current\_server]

    for table in assigned\_tables:

        print(f"Table {table.table\_num}")

    table\_num = int(input("Enter the table number: "))

    table = next((table for table in table\_list if table.table\_num == table\_num and table.server == current\_server), None)

    if table:

        bill, total = table.prepare\_bill()

        print("Bill:")

        print(bill)

        file\_name = input("Enter the file name to save the bill: ")

        with open(file\_name, 'w') as file:

            file.write(bill)

        print("Bill successfully prepared!")

        return total

    else:

        print("Invalid table number or the table is not assigned to you.")

        return 0.0

def complete\_sale(current\_server, table\_list, current\_sale):

    print("Tables assigned to you:")

    assigned\_tables = [table for table in table\_list if table.server == current\_server]

    for table in assigned\_tables:

        print(f"Table {table.table\_num}")

    table\_num = int(input("Enter the table number: "))

    table = next((table for table in table\_list if table.table\_num == table\_num and table.server == current\_server), None)

    if table:

        if len(table.orders) > 0:

            total = prepare\_bill(current\_server, table\_list)

            current\_sale.total\_sale += total

            table.clear\_table()

            print("Sale completed successfully!")

        else:

            print("No orders found for the table.")

    else:

        print("Invalid table number or the table is not assigned to you.")

def cash\_up(current\_sale):

    print("Total sales: R", current\_sale.total\_sale)

    choice = input("Do you want to clear the daily total? (y/n): ")

    if choice.lower() == "y":

        current\_sale.total\_sale = 0.0

        print("Daily total cleared.")

def point\_of\_sale():

    login\_file = "Login.txt"

    menu\_file = "Menu.txt"

    login\_cred = read\_login\_cred(login\_file)

    menu\_items = read\_menu(menu\_file)

    table\_list = [Table(i + 1) for i in range(6)]

    current\_sale = Sale()

    current\_server = input("Username: ")

    passwrd = input("passwrd: ")

    if current\_server in login\_cred and login\_cred[current\_server] == passwrd:

        print("Login successful!")

        while True:

            display\_menu()

            choice = int(input("Enter your choice: "))

            if choice == 1:

                assign\_table(current\_server, table\_list)

            elif choice == 2:

                change\_customers(current\_server, table\_list)

            elif choice == 3:

                add\_to\_order(current\_server, table\_list, menu\_items)

            elif choice == 4:

                prepare\_bill(current\_server, table\_list)

            elif choice == 5:

                complete\_sale(current\_server, table\_list, current\_sale)

            elif choice == 6:

                cash\_up(current\_sale)

            elif choice == 0:

                print("Logged out successfully!")

                break

            else:

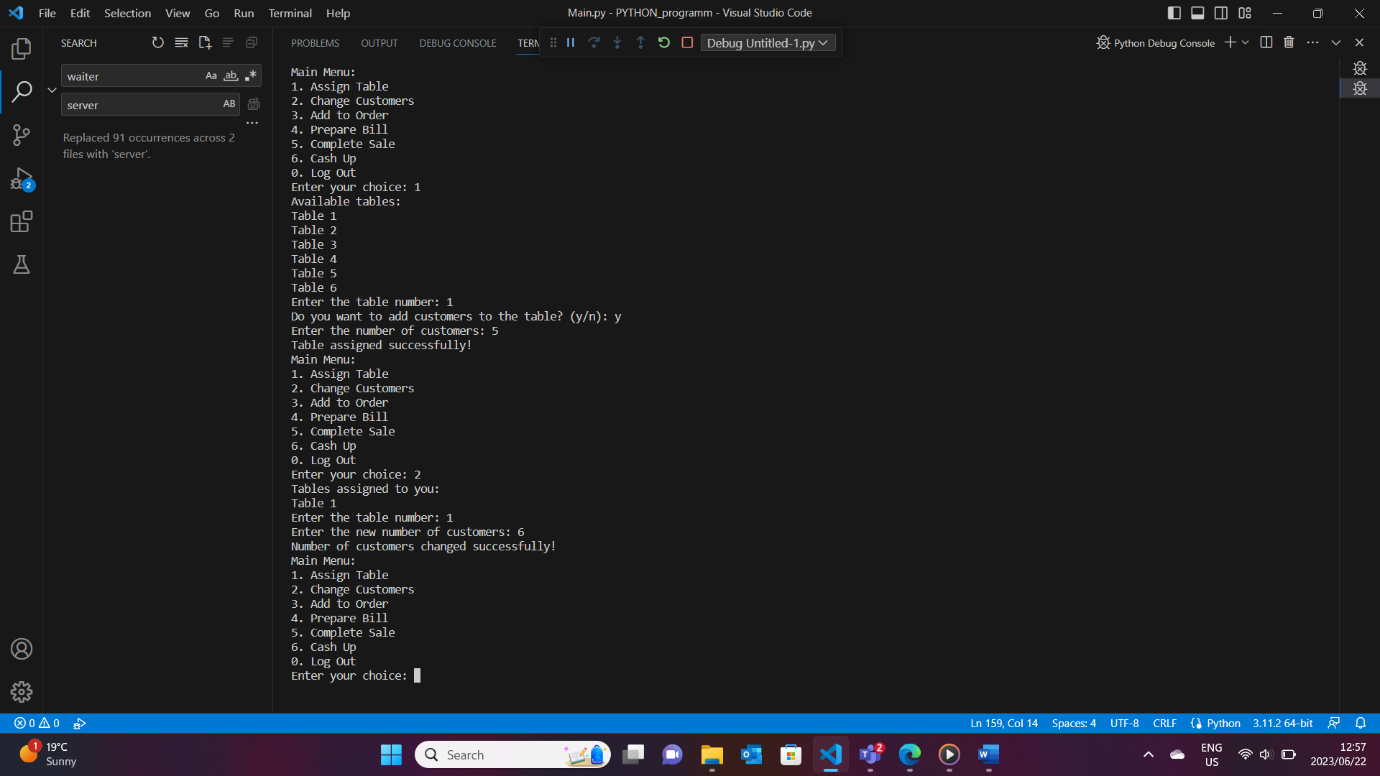
                print("Invalid choice. Please try again.")

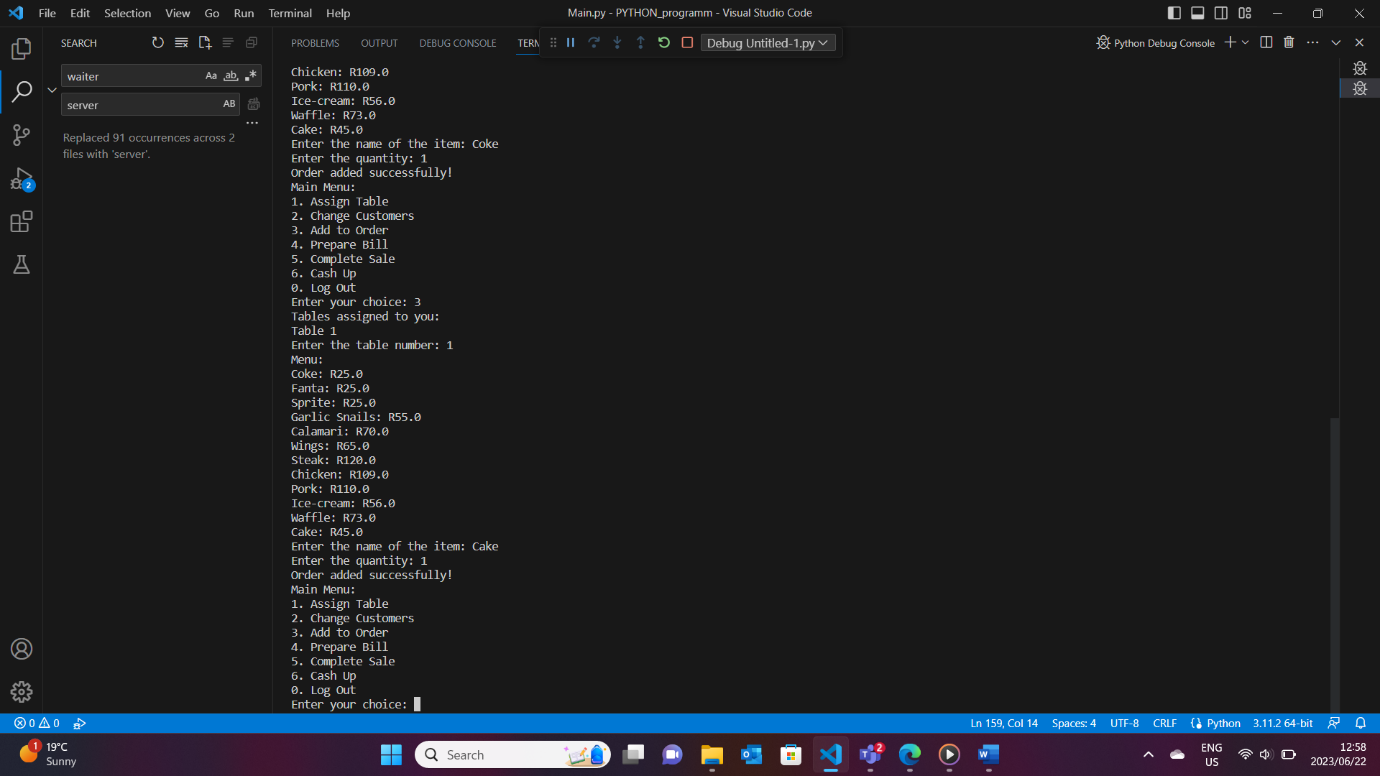
    else:

        print("Invalid username or passwrd. Login failed.")

point\_of\_sale()

# Output:

****

****

**A picture containing screenshot, text, software, multimedia software

Description automatically generated**

**A picture containing screenshot, text, software, multimedia software

Description automatically generated**

**A picture containing screenshot, text, software, multimedia software

Description automatically generated**

