Lab:08.

Q1: Build classes for Product, ShoppingCart, and Customer. Implement methods to add products to the cart, display the cart contents, and calculate the total cost.

Program:

```
class Product:
    def __init__(self, product_id, name, price):
        self.product_id = product_id
        self.name = name
        self.price = price

    def __str__(self):
        return f"{self.name} (ID: {self.product_id}), Price: ${self.price:.2f}"


class ShoppingCart:
    def __init__(self):
        self.cart = []

    def add_product(self, product, quantity=1):
        for item in self.cart:
            if item["product"] == product:
                item["quantity"] += quantity
                print(f"Added {quantity} {product.name}(s) to the cart.")
                return

        self.cart.append({"product": product, "quantity": quantity})
        print(f"Added {quantity} {product.name}(s) to the cart.")
    def display_cart(self):
        if not self.cart:
            print("The cart is empty.")
        else:
            print("Shopping Cart:")
            for item in self.cart:
                product = item["product"]
                quantity = item["quantity"]
                print(f"{product} x{quantity}")

    def calculate_total_cost(self):
        total_cost = sum(item["product"].price * item["quantity"] for item in self.cart)
        return total_cost


class Customer:
    def __init__(self, customer_id, name):
        self.customer_id = customer_id
        self.name = name
        self.shopping_cart = ShoppingCart()

    def checkout(self):
        total_cost = self.shopping_cart.calculate_total_cost()
        print(f"{self.name}'s Shopping Cart Total: ${total_cost:.2f}")


if __name__ == "__main__":
```

```
customer = Customer(1, input("Enter your name: "))

while True:
    print("\nShopping Options:")
    print("1. Add Product to Cart")
    print("2. Display Cart")
    print("3. Calculate Total Cost")
    print("4. Checkout")
    print("5. Exit")

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

    if choice == "1":
        product_id = int(input("Enter the product ID: "))
        name = input("Enter the product name: ")
        price = float(input("Enter the product price: "))
        quantity = int(input("Enter the quantity: "))
        product = Product(product_id, name, price)
        customer.shopping_cart.add_product(product, quantity)

    elif choice == "2":
        customer.shopping_cart.display_cart()

    elif choice == "3":
        total_cost = customer.shopping_cart.calculate_total_cost()
        print(f"Total Cost: ${total_cost:.2f}")

    elif choice == "4":
        customer.checkout()

    elif choice == "5":
        print("Exiting the program. Goodbye!")
        break

    else:
        print("Invalid choice. Please enter a number between 1 and 5.")
```

Q2: Design classes for Blog, Post, and Author. Include methods to add posts to a blog, display posts by a specific author, and display the latest posts.

Program:

```
from datetime import datetime
class Author:
    def __init__(self, author_id, name):
        self.author_id = author_id
        self.name = name

    def __str__(self):
        return f"Author ID: {self.author_id}, Name: {self.name}"



class Post:
    def __init__(self, post_id, title, content, author, timestamp=None):
        self.post_id = post_id
        self.title = title
        self.content = content
        self.author = author
        self.timestamp = timestamp or datetime.now()

    def __str__(self):
        return f"Post ID: {self.post_id}\nTitle: {self.title}\nContent:
{self.content}\nAuthor: {self.author.name}\nTimestamp: {self.timestamp}"

class Blog:
    def __init__(self):
        self.posts = []

    def add_post(self, post):
        self.posts.append(post)
        print(f"Post '{post.title}' added to the blog.")

    def display_posts_by_author(self, author):
        author_posts = [post for post in self.posts if post.author == author]
        if author_posts:
            print(f"Posts by {author.name}:")
            for post in author_posts:
                print(post)
        else:
            print(f"No posts found by {author.name}.")

    def display_latest_posts(self, num_posts=5):
        if not self.posts:
            print("No posts in the blog.")
        else:
            latest_posts = sorted(self.posts, key=lambda x: x.timestamp,
reverse=True)[:num_posts]
            print(f"Latest {num_posts} Posts:")
            for post in latest_posts:
                print(post)



# Example Usage with User Input:
if __name__ == "__main__":
    # Creating an instance of Blog
    blog = Blog()

    while True:
        print("\nBlog Menu:")
        print("1. Add Post to Blog")
        print("2. Display Posts by Author")
        print("3. Display Latest Posts")
        print("4. Exit")

        choice = input("Enter your choice (1-4): ")

        if choice == "1":
            author_name = input("Enter the author's name: ")
            post_title = input("Enter the post title: ")
            post_content = input("Enter the post content: ")

            author = Author(len(blog.posts) + 1, author_name)
            post = Post(len(blog.posts) + 1, post_title, post_content, author)
            blog.add_post(post)

        elif choice == "2":
            author_name = input("Enter the author's name: ")
            author_to_display = next((author for post in blog.posts if
post.author.name == author_name), None)
            if author_to_display:
                blog.display_posts_by_author(author_to_display)
            else:
                print("Author not found.")

        elif choice == "3":
            num_posts = int(input("Enter the number of latest posts to display:
"))
            blog.display_latest_posts(num_posts)

        elif choice == "4":
            print("Exiting the program. Goodbye!")
            break

        else:
```

```
print("Invalid choice. Please enter a number between 1 and 4.")
```