NAME LAIBA BIBI

BCS2310-40

ASSIGNMENT #3

//CODE1

#include <iostream>

using namespace std;

// Abstract base class

class Shape {

public:

// Pure virtual function

virtual double calculateArea() const = 0;

//destructor

virtual ~Shape() {}

};

//derived class

class Circle : public Shape {

private:

double radius;

public:

Circle(double r) : radius(r) {}

double calculateArea() const override {

return 3.14159 \* radius \* radius;

}

};

// CODE 2 QUESTION 2

class Rectangle : public Shape {

private:

double width;

double height;

public:

Rectangle(double w, double h) : width(w), height(h) {}

double calculateArea() const override {

return width \* height;

}

};

int main() {

// Create instances /objects of Circle and Rectangle

Circle circle(5.0);

Rectangle rectangle(4.0, 6.0);

cout << "Area of the circle: " << circle.calculateArea() << endl;

cout << "Area of the rectangle: " << rectangle.calculateArea() << endl;

return 0;

}

//code 2

#include <iostream>

#include <string>

#include <vector>

using namespace std;

class Product {

private:

int productId;

string productName;

double price;

public:

Product(int id, const string& name, double p) : productId(id), productName(name), price(p) {}

void displayProductDetails() const {

cout << "Product ID: " << productId << ", Product Name: " << productName << ", Price: $" << price << endl;

}

double getPrice() const {

return price;

}

};

class ShoppingCart {

private:

vector<Product\*> products;

public:

void addProduct(Product\* product) {

products.push\_back(product);

cout << "Added product to the shopping cart." << endl;

}

void displayAllProducts() const {

cout << "Products in the shopping cart:" << endl;

for (const auto& product : products) {

product->displayProductDetails();

}

}

double calculateTotalCost() const {

double totalCost = 0.0;

for (const auto& product : products) {

totalCost += product->getPrice();

}

return totalCost;

}

};

class User {

private:

int userId;

ShoppingCart\* shoppingCart; // Aggregation relationship

public:

User(int id) : userId(id), shoppingCart(nullptr) {}

void displayUserDetails() const {

cout << "User ID: " << userId << endl;

if (shoppingCart != nullptr) {

cout << "User has a shopping cart." << endl;

} else {

cout << "User does not have a shopping cart." << endl;

}

}

void associateShoppingCart(ShoppingCart\* cart) {

shoppingCart = cart; // Association relationship

cout << "User associated with a shopping cart." << endl;

}

// Destructor

~User() {

delete shoppingCart;

}

};

int main() {

Product\* laptop = new Product(1, "Laptop", 999.99);

Product\* phone = new Product(2, "Smartphone", 499.99);

ShoppingCart\* cart = new ShoppingCart();

cart->addProduct(laptop);

cart->addProduct(phone);

User\* user = new User(1001);

user->associateShoppingCart(cart);

// Display user details

user->displayUserDetails();

// Display products

user->associateShoppingCart(cart);

cart->displayAllProducts();

double totalCost = cart->calculateTotalCost();

cout << "Total cost of the products: $" << totalCost << endl;

delete user;

delete laptop;

delete phone;

return 0;

}