**Лабораторна робота №18**

**Тема: Створення ПМ на основі представлень архітектури**

**2. Порядок виконання роботи**

**І.** Вивчення коротких відомостей.

**ІІ.** Виконання завдання:

**Завдання:**

Створення ПМ на основі представлень архітектури

**Варіанти завдань:**

**1 варіант**

1. 1. Моделювання роботи магазину.
2. Вхідні данні:
3. клас Product;
   * + 1. клас Description;
   1. клас Customer;
4. клас Order;
5. клас Employee;
6. Методи: додавання, видалення, редагування інформації.

**ІІІ.** Складіть звіт, приєднавши отримані результати.

**Вимоги до звіту.**

Звіт за результатами роботи повинен мати в своєму складі:

завдання;

тестовий приклад;

аналіз результатів;

відпрацьовані артефакти (діаграми, тексти програм інші документи).

Виконання

Діаграма

Class Main є головним класом ,який відповідає за запуск і роботу програми

Class Order відповідає за створення замовлення і передачу його в Main

Order в свою чергу отримує інформацію з класів Employee \ Customer\Product

Class Employee передає в клас Order інформацію про продавця

Class Customer передає в клас Order інформацію про замовника і його баланс

Class Product додається в кошик ордеру

Class Description описує клас Product

КОД

class Description

{

public string Title { get; private set; }

public string Text { get; private set; }

public string ExpirationDate { get; private set; }

public string Category { get; private set; }

public Description(string Title, string Text, string ExpirationDate, string Category)

{

this.Title = Title;

this.Text = Text;

this.ExpirationDate = ExpirationDate;

this.Category = Category;

}

public void ChangeTitle(string Title)

{

this.Title = Title;

}

public void ChangeDescription(string Text)

{

this.Text = Text;

}

public void ChangeExpirationDate(string ExpirationDate)

{

this.ExpirationDate = ExpirationDate;

}

public void ChangeCategory(string Category)

{

this.Category = Category;

}

}

class Product

{

public int Id { get; private set; }

public string Name { get; private set; }

public double Price { get; private set; }

public Description Description { get; private set; }

public Product(int Id, string Name, double Price, Description Description)

{

this.Id = Id;

this.Name = Name;

this.Price = Price;

this.Description = Description;

}

public void ChangeId(int Id)

{

this.Id = Id;

}

public void ChangeName(string Name)

{

this.Name = Name;

}

public void ChangePrice(int Price)

{

this.Price = Price;

}

public void ChangeDescription(string Title, string Text, string ExpirationDate, string Category)

{

this.Description = new Description(Title, Text, ExpirationDate, Category);

}

public void ChangeDescription(Description Description)

{

this.Description = Description;

}

}

class Order

{

public int Id { get; private set; }

public double TotalPrice { get; private set; }

public List<Product> ProductList { get; private set; }

public Order(int Id)

{

this.Id = Id;

this.TotalPrice = 0.0;

this.ProductList = new List<Product>();

}

public void ChangeId(int Id)

{

this.Id = Id;

}

public void ChangeTotalPrice(double TotalPrice)

{

this.TotalPrice = TotalPrice;

}

public void AddToList(Product Product)

{

this.ProductList.Add(Product);

this.TotalPrice += Product.Price;

}

public void DeleteToList(Product Product)

{

this.ProductList.Remove(Product);

this.TotalPrice -= Product.Price;

}

}

class Customer

{

public int Id { get; private set; }

public string Name { get; private set; }

public string Surname { get; private set; }

public string Telephone { get; private set; }

public double PersonalAccount { get; private set; }

public Customer(int Id, string Name, string Surname, string Telephone, double PersonalAccount)

{

this.Id = Id;

this.Name = Name;

this.Surname = Surname;

this.Telephone = Telephone;

this.PersonalAccount = PersonalAccount;

}

public void BuyOrder(Order Order)

{

if (this.PersonalAccount >= Order.TotalPrice)

{

this.PersonalAccount -= Order.TotalPrice;

System.Console.WriteLine($"Operation #{Order.Id} for customer #{this.Id} complated successfully!", 0, 1);

}

else

{

System.Console.WriteLine($"Operation #{Order.Id} for customer #{this.Id} failed!", 0, 1);

}

}

}

class Employee

{

public int Id { get; private set; }

public string Name { get; private set; }

public string Surname { get; private set; }

public string Telephone { get; private set; }

public string HomeAddress { get; private set; }

public double Salary { get; private set; }

public List<Order> OrderList { get; private set; }

public Employee(int Id, string Name, string Surname, string Telephone, string HomeAddress, double Salary)

{

this.Id = Id;

this.Name = Name;

this.Surname = Surname;

this.Telephone = Telephone;

this.HomeAddress = HomeAddress;

this.Salary = Salary;

this.OrderList = new List<Order>();

}

public void ChangeId(int Id)

{

this.Id = Id;

}

public void ChangeName(string Name)

{

this.Name = Name;

}

public void ChangeSurname(string Surname)

{

this.Surname = Surname;

}

public void ChangeTelephone(string Telephone)

{

this.Telephone = Telephone;

}

public void ChangehomeAddress(string HomeAddress)

{

this.HomeAddress = HomeAddress;

}

public void ChangeSalary(double Salary)

{

this.Salary = Salary;

}

public void AddToList(Order Order)

{

this.OrderList.Add(Order);

}

public void DeleteToList(int Id)

{

this.OrderList.RemoveAt(Id);

}

}

class Program

{

static void Main(string[] args)

{

Employee employee = new Employee(21018771, "John", "Kart", "380953454532", "Kyiv", 2000.0);

List<Customer> customerList = new List<Customer>();

Customer customer1 = new Customer(72641214, "Tom", "Jerry", "380542121371", 500.0);

Customer customer2 = new Customer(92302458, "John", "Konor", "380512838313", 100.0);

customerList.Add(customer1);

customerList.Add(customer2);

foreach(Customer customer in customerList)

{

System.Console.WriteLine($"Customer #{customer.Id} personal account before buy products: {customer.PersonalAccount}", 0, 1);

}

Order order1 = new Order(12320194);

Order order2 = new Order(82401924);

Order order3 = new Order(98738901);

Product milk = new Product(45004127, "Milk", 20.0, new Description("Milk", "The best milk", "21.12.2021", "latex"));

Product meet = new Product(23019054, "Meet", 45.0, new Description("Pork", "The best meet", "03.01.2022", "meat"));

Product apple = new Product(35201127, "Apple", 5.0, new Description("Apple", "The best apple", "19.12.2021", "fruit"));

Product bread = new Product(20910234, "Bread", 15.0, new Description("Bread", "The best bread", "16.12.2021", "bread"));

order1.AddToList(milk);

order1.AddToList(meet);

order1.AddToList(apple);

order2.AddToList(bread);

System.Console.WriteLine("Order list:");

System.Console.WriteLine($"Order1 #{order1.Id} total price: {order1.TotalPrice}", 0, 1);

System.Console.WriteLine($"Order2 #{order2.Id} total price: {order2.TotalPrice}", 0, 1);

System.Console.WriteLine($"Order3 #{order3.Id} total price: {order3.TotalPrice}", 0, 1);

order1.DeleteToList(meet);

order3.AddToList(meet);

System.Console.WriteLine("Order list after change:");

System.Console.WriteLine($"Order1 #{order1.Id} total price: {order1.TotalPrice}", 0, 1);

System.Console.WriteLine($"Order2 #{order2.Id} total price: {order2.TotalPrice}", 0, 1);

System.Console.WriteLine($"Order3 #{order3.Id} total price: {order3.TotalPrice}", 0, 1);

employee.AddToList(order1);

employee.AddToList(order2);

employee.AddToList(order3);

customer1.BuyOrder(employee.OrderList[0]);

customer1.BuyOrder(employee.OrderList[2]);

customer2.BuyOrder(employee.OrderList[1]);

foreach (Customer customer in customerList)

{

System.Console.WriteLine($"Customer #{customer.Id} personal account after buy products: {customer.PersonalAccount}", 0, 1);

}

employee.OrderList.Clear();

}

}