**«Платформа Microsoft .NET та мова програмування C#»**

**Завдання 1.**

Розробити архітектуру класів ієрархії товарів для розробки системи управління потоками товарів для дистриб'юторської компанії. Прописати членів класів.

Повинні бути передбачені різні типи товарів, у тому числі:

• побутова хімія,

• продукти харчування.

Передбачити інтерфейси:

• продукти, що швидко псуються,

• лікеро-горілчані продукти та тютюнові вироби (акцизні продукти),

• легкозаймисті товари,

• товари, що б'ються.

Передбачити класи керування потоком товарів (прийшло, реалізовано, списано, передано).

Реалізувати власні класи винятків:

• «немає в наявності»,

• "закінчився термін придатності",

• "бракований товар".

using System;

using System.Collections;

using System.Collections.Generic;

namespace test11

{

class NotAvailable:Exception

{

string code;

public NotAvailable()

{

code = "0X4X20X12X8";

}

public override string ToString()

{

return "Немає в наявності\nКод помилки: " + code;

}

}

class Expired : Exception

{

string code;

public Expired()

{

code = "0X1X618X11X99";

}

public override string ToString()

{

return "Закінчився термін придатності\nКод помилки: " + code;

}

}

class Defective : Exception

{

string code;

public Defective()

{

code = "12X98X44X10X1X1";

}

public override string ToString()

{

return "Бракований товар\nКод помилки: " + code;

}

}

interface ISpoilsQuickly

{

public bool ISSpoilsQuickly { set; get; }

}

interface IAlcoholicAndTobacco

{

public bool ISAlcoholicOrTobacco { set; get; }

}

interface IFlammableGoods//легкозаймисті

{

public bool IsFlammableGoods { set; get; }

}

interface IPerishableGoods//товари що б'ються

{

public bool IsPerishableGoods { set; get; }

}

abstract class Product

{

string name;

string code;

DateTime ExpirationDate;

bool IsDefective;

string packaging;

string manufacturer\_of\_the\_product;

public string Name

{

get { return name; }

set { name = value; }

}

public string Code

{

get { return code; }

set { code = value; }

}

public DateTime EXpirationDate

{

get { return ExpirationDate; }

set { ExpirationDate = value; }

}

public bool ISDefective

{

get { return IsDefective; }

set { IsDefective = value; }

}

public string Packaging

{

get { return packaging; }

set { packaging = value; }

}

public string Manufacturer\_of\_the\_product

{

get { return manufacturer\_of\_the\_product; }

set { manufacturer\_of\_the\_product = value; }

}

public Product()

{

name = "unknown";

ExpirationDate = new DateTime(1, 1, 1);

IsDefective = false;

packaging= "unknown";

manufacturer\_of\_the\_product= "unknown";

code= "unknown";

}

public Product(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code)

{

this.name = name;

this.ExpirationDate = ExpirationDate;

this.IsDefective = IsDefective;

this.packaging = packaging;

this.manufacturer\_of\_the\_product = manufacturer\_of\_the\_product;

this.code = code;

}

public virtual void PrintInfo()

{

Console.WriteLine(this.name);

Console.WriteLine("Придатний до: "+this.EXpirationDate.ToShortDateString());

if(IsDefective)

Console.WriteLine("Продукт бракований");

else

Console.WriteLine("Новий");

Console.WriteLine("Упаковка: "+ packaging);

Console.WriteLine("Код продукту: " + code);

Console.WriteLine("Виробник: " + manufacturer\_of\_the\_product);

}

}

class Meat:Product, ISpoilsQuickly

{

bool ispoilsQuickly;

public bool ISSpoilsQuickly

{

set { ispoilsQuickly = value; }

get { return ispoilsQuickly; }

}

public Meat(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code) :base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product, code)

{

ispoilsQuickly = false;

}

public Meat(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code, bool ispoilsQuickly) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product, code)

{

this.ispoilsQuickly = ispoilsQuickly;

}

public override void PrintInfo()

{

base.PrintInfo();

if (ispoilsQuickly)

Console.WriteLine("Продукт швидко псується");

}

}

class Liqueur : Product, IAlcoholicAndTobacco

{

bool isalcoholicAndTobacco;

public bool ISAlcoholicOrTobacco

{

set { isalcoholicAndTobacco = value; }

get { return isalcoholicAndTobacco; }

}

public Liqueur(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product, code)

{

isalcoholicAndTobacco = false;

}

public Liqueur(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code, bool isalcoholicAndTobacco) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product, code)

{

this.isalcoholicAndTobacco = isalcoholicAndTobacco;

}

public override void PrintInfo()

{

base.PrintInfo();

if (isalcoholicAndTobacco)

Console.WriteLine("Цей продукт є акцизним");

}

}

class HouseholdChemicals : Product, IFlammableGoods

{

bool isFlammableGoods;

public bool IsFlammableGoods

{

set { isFlammableGoods = value; }

get { return isFlammableGoods; }

}

public HouseholdChemicals(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product,string code) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product,code)

{

isFlammableGoods = false;

}

public HouseholdChemicals(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code, bool isFlammableGoods) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product,code)

{

this.isFlammableGoods = isFlammableGoods;

}

public override void PrintInfo()

{

base.PrintInfo();

if (isFlammableGoods)

Console.WriteLine("Цей продукт є легкозаймистим");

}

}

class Juice : Product, IPerishableGoods

{

bool isPerishableGoods;

public bool IsPerishableGoods

{

set { isPerishableGoods = value; }

get { return isPerishableGoods; }

}

public Juice(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product, code)

{

isPerishableGoods = false;

}

public Juice(string name, DateTime ExpirationDate, bool IsDefective, string packaging, string manufacturer\_of\_the\_product, string code, bool isPerishableGoods) : base(name, ExpirationDate, IsDefective, packaging, manufacturer\_of\_the\_product, code)

{

this.isPerishableGoods = isPerishableGoods;

}

public override void PrintInfo()

{

base.PrintInfo();

if (isPerishableGoods)

Console.WriteLine("Цей товар легко б'ється");

}

}

class Realize

{

List<Product> products;

public Realize(List<Product> products)

{

this.products = products;

}

public void DoRealize(string code,DateTime dateTime)

{

int index = -1;

for(int i=0;i<products.Count;i++)

{

if (string.Compare(products[i].Code, code) == 0 )

{

if(products[i].ISDefective==true)

{

throw new Defective();

}

if (dateTime > products[i].EXpirationDate)

{

throw new Expired();

//throw

}

index = i;

break;

}

}

if (index != -1)

{

products.RemoveAt(index);

Console.WriteLine("Товар реалізовано");

}

else

{

throw new NotAvailable();

}

}

}

class ImportOfGoods

{

List<Product> products;

public ImportOfGoods(List<Product> products)

{

this.products = products;

}

public void DoImportOfGoods(Product product)

{

products.Add(product);

Console.WriteLine("Товар додано до складу");

}

}

class WritingOffGoods

{

List<Product> products;

public WritingOffGoods(List<Product> products)

{

this.products = products;

}

public void DoWritingOffGoods(string code)

{

int index = -1;

for (int i = 0; i < products.Count; i++)

{

if (string.Compare(products[i].Code, code) == 0)

{

index = i;

break;

}

}

if (index != -1)

{

products.RemoveAt(index);

Console.WriteLine("Товар списано");

}

else

{

throw new NotAvailable();

}

}

}

class Transfer

{

List<Product> products;

public Transfer(List<Product> products)

{

this.products = products;

}

public void DoTransfer(string code)

{

int index = -1;

for (int i = 0; i < products.Count; i++)

{

if (string.Compare(products[i].Code, code) == 0)

{

index = i;

break;

}

}

if (index != -1)

{

products.RemoveAt(index);

Console.WriteLine("Товар передано");

}

else

{

throw new NotAvailable();

}

}

}

class WarehouseOfGoods

{

List<Product> products;

Realize realize;

ImportOfGoods importOfGoods;

WritingOffGoods writingOffGoods;

Transfer transfer;

public WarehouseOfGoods()

{

products = new List<Product>();

realize = new Realize(products);

importOfGoods = new ImportOfGoods(products);

writingOffGoods = new WritingOffGoods(products);

transfer = new Transfer(products);

}

public void DoRealize(string code, DateTime dateTime)

{

realize.DoRealize(code, dateTime);

}

public void DoImportOfGoods(Product product)

{

importOfGoods.DoImportOfGoods(product);

}

public void DoWritingOffGoods(string code)

{

writingOffGoods.DoWritingOffGoods(code);

}

public void DoTransfer(string code)

{

transfer.DoTransfer(code);

}

public void PrintListProducts()

{

Console.WriteLine("-------------------------------------");

for (int i=0;i<products.Count;i++)

{

products[i].PrintInfo();

Console.WriteLine("-------------------------------------");

}

}

}

class Program

{

public static void Main()

{

System.Console.OutputEncoding = System.Text.Encoding.UTF8;

WarehouseOfGoods warehouseOfGoods = new WarehouseOfGoods();

try

{

warehouseOfGoods.DoImportOfGoods(new Meat("Куряче філе", new DateTime(2022, 11, 20), false, "Вакуумна", "Наша Ряба", "123451", false));

warehouseOfGoods.DoImportOfGoods(new Liqueur("Sheridan's", new DateTime(2024, 1, 10), false, "Пластикова бутилка", "Ірландiя", "1512435", true));

warehouseOfGoods.DoImportOfGoods(new HouseholdChemicals("Fairy", new DateTime(2023, 7, 28), true, "Пластикова бутилка", "Україна", "328192", true));

warehouseOfGoods.DoImportOfGoods(new Juice("Galicia яблучно-морквяний", new DateTime(2022, 12, 15), false, "Скляна бутилка", "Україна", "52346232", true));

warehouseOfGoods.PrintListProducts();

warehouseOfGoods.DoRealize("1512435", new DateTime(2022, 2, 10));//new DateTime це дата коли реалізовується продукт

//warehouseOfGoods.DoRealize("123451", new DateTime(2023, 11, 20));

//warehouseOfGoods.DoRealize("328192", new DateTime(2022, 12, 11));

//warehouseOfGoods.DoRealize("fdsfds", new DateTime(2022, 12, 11));

warehouseOfGoods.PrintListProducts();

}

catch(NotAvailable e)

{

Console.WriteLine(e);

}

catch (Expired e)

{

Console.WriteLine(e);

}

catch (Defective e)

{

Console.WriteLine(e);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

}