Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

**Звіт**

з лабораторної роботи №5 з дисципліни

«Основи програмування – 2. Методології програмування»

«Успадкування та поліморфізм»   
Варіант 22

Виконав студент ІП-13, Музичук Віталій Андрійович

(шифр, прізвище, ім'я, по батькові)

Перевірила Вєчерковська Анастасія Сергіївна

( прізвище, ім'я, по батькові)

Київ 2022

**Лабораторна робота 4**

**Перевантаження операторів**

**Мета** – вивчити механізми створення класів з використанням класів та об’єктів.

**Варіант 22**

**Завдання:**

Створити клас “Товар”, який містить назву, дату виготовлення, ціну, кількість одиниць, а також метод обчислення сумарної вартості товару. На його основі створити класи-нащадки “Промисловий товар”, що додатково зберігає умови транспортування, місце знаходження товару (на складі, в торговому залі) та “Харчовий продукт”, який додатково містить термін зберігання продукту. Створити n номенклатур промислових товарів та m номенклатур харчових продуктів. Визначити загальну вартість харчових продуктів термін зберігання яких закінчився, і загальну вартість промислових товарів, які знаходяться на складі.

*1. Виконання завдання на мові С++:*

**// Lab\_5.cpp**

#include "Goods.h"

int main()

{

cout << "How many industrial products do you want create: ";

int numInd; cin >> numInd;

IndustrialGoods\* indGoods = createInd(numInd);

cout << "How many eating products do you want create: ";

int numEat; cin >> numEat;

EatingGoods\* eatGoods = createEat(numEat);

struct tm current\_time;

time\_t t = time(0);

localtime\_s(&current\_time, &t);

TDate currentTime(current\_time.tm\_mday, current\_time.tm\_mon + 1, 1900 + current\_time.tm\_year);

int full\_price = 0;

for (int i = 0; i < numEat; i++) {

if (eatGoods[i].isSpoiledGood(currentTime)) {

full\_price += eatGoods[i].get\_sum();

}

}

cout << "The total price of all spoiled products is: " << full\_price << endl;

full\_price = 0;

for (int i = 0; i < numInd; i++) {

if (indGoods[i].get\_conditions() == IN\_STOCK) {

full\_price += indGoods[i].get\_sum();

}

}

cout << "The total price of all products that store IN\_STOCK: " << full\_price << endl;

return 0;

}

**//Goods.cpp**

#include "Goods.h"

// Реалізація класу для збереження дати

int TDate::monthDate[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

TDate::TDate(int day1 = 1, int month1 = 1, int year1 = 1901) {

if (year1 < 1900 || year1 > 2022) {

cerr << "Incorrect enter year";

exit(0);

}

if (month1 <= 0 || month1 > 12) {

cerr << "Invalid enter month";

exit(0);

}

if (day1 <= 0 || day1 > monthDate[month1 - 1]) {

cerr << "Invalid entered day";

exit(0);

}

this->day = day1;

this->month = month1;

this->year = year1;

}

void TDate::addDay(int day) {

if (day < 0) {

cerr << "Incorrected day";

exit(0);

}

this->day += day;

while (true) {

if (this->day > monthDate[this->month - 1]) {

this->day -= monthDate[this->month - 1];

this->month += 1;

}

else break;

}

}

// Реалізація базового класу

Goods::Goods() {

name = "noName";

date;

price = 0.0;

number = 0;

}

Goods::Goods(string name1, TDate date1, float price1, int number1) : name(name1), date(date1) {

if (price1 < 0) {

cerr << "Incorrect entered price";

exit(0);

}

if (number1 <= 0) {

cerr << "Invalid number of products";

exit(0);

}

this->price = price1;

this->number = number1;

}

float Goods::get\_sum() {

return this->price \* this->number;

}

// Реалізація похідного класу для збереження промислових товарів

IndustrialGoods::IndustrialGoods(): Goods(), conditions(IN\_STOCK) {}

IndustrialGoods::IndustrialGoods(string name1, TDate date1, float price1, int number1, Transport conditions1) :

Goods(name1, date1, price1, number1), conditions(conditions1) {}

Transport IndustrialGoods::get\_conditions() {

return this->conditions;

}

// Реалізація похідного класу для збереження харчових товарів

EatingGoods::EatingGoods() : Goods(), expiration(0) {}

EatingGoods::EatingGoods(string name1, TDate date1, float price1, int number1, int expiration1) :

Goods(name1, date1, price1, number1){

if (expiration1 < 0) {

cerr << "Incorrected expiration date";

exit(0);

}

this->expiration = expiration1;

}

bool EatingGoods::isSpoiledGood(TDate& currentDate) {

TDate expirationDate = this->date;

expirationDate.addDay(this->expiration);

int cuurentDays = currentDate.year \* 365 + (currentDate.month - 1) \* 31 + currentDate.day;

int expirationDays = expirationDate.year \* 365 + (expirationDate.month - 1) \* 31 + expirationDate.day;

if (cuurentDays > expirationDays) {

return true;

}

else if (expirationDays >= cuurentDays) {

return false;

}

}

// Допоміжні функції

IndustrialGoods\* createInd(int numInd) {

IndustrialGoods\* indGoods = new IndustrialGoods[numInd];

cout << "Add information about industrial product" << endl;

string name, dateLine; float price; int num, cond; int dates[3];

for (int i = 0; i < numInd; i++) {

cout << "Name product: ";

cin.ignore(3200, '\n');

getline(cin, name);

cout << "Enter date in format[dd.mm.yyyy]: ";

getline(cin, dateLine);

for (int j = 0; j < 3; j++)

{

if (j == 2)

dates[j] = stoi(dateLine);

else {

int pos = dateLine.find('.');

dates[j] = stoi(dateLine.substr(0, pos));

dateLine.erase(0, pos + 1);

}

}

cout << "Enter price: ";

cin >> price;

cout << "Enter number of products: ";

cin >> num;

cout << "Enter a conditions of transoprt(0 - IN\_STOCK, 1 - IN\_TRADING\_HALL): ";

cin >> cond;

indGoods[i] = IndustrialGoods(name, TDate(dates[0], dates[1], dates[2]), price, num, static\_cast<Transport>(cond));

cout << endl;

}

return indGoods;

}

EatingGoods\* createEat(int numEat) {

EatingGoods\* eatGoods = new EatingGoods[numEat];

cout << "Add information about eating product" << endl;

string name, dateLine; float price; int num, expiriation; int dates[3];

for (int i = 0; i < numEat; i++) {

cout << "Name product: ";

cin.ignore(3200, '\n');

getline(cin, name);

cout << "Enter date in format[dd.mm.yyyy]: ";

getline(cin, dateLine);

for (int j = 0; j < 3; j++)

{

if (j == 2)

dates[j] = stoi(dateLine);

else {

int pos = dateLine.find('.');

dates[j] = stoi(dateLine.substr(0, pos));

dateLine.erase(0, pos + 1);

}

}

cout << "Enter price: ";

cin >> price;

cout << "Enter number of products: ";

cin >> num;

cout << "Enter a expiration date: ";

cin >> expiriation;

eatGoods[i] = EatingGoods(name, TDate(dates[0], dates[1], dates[2]), price, num, expiriation);

cout << endl;

}

return eatGoods;

}

**// vector.h**

#pragma once

#include <iostream>

#include <string>

using namespace std;

class TDate {

public:

int day;

int month;

int year;

static int monthDate[12];

TDate(int day1, int month1, int year1);

void addDay(int);

};

enum Transport {

IN\_STOCK,

IN\_TRADING\_HALL

};

class Goods {

protected:

string name;

TDate date;

float price;

int number;

Goods();

Goods(string, TDate, float, int);

public:

float get\_sum();

};

class IndustrialGoods : public Goods {

Transport conditions;

public:

IndustrialGoods();

IndustrialGoods(string, TDate, float, int, Transport);

Transport get\_conditions();

};

class EatingGoods : public Goods {

int expiration;

public:

EatingGoods();

EatingGoods(string, TDate, float, int, int);

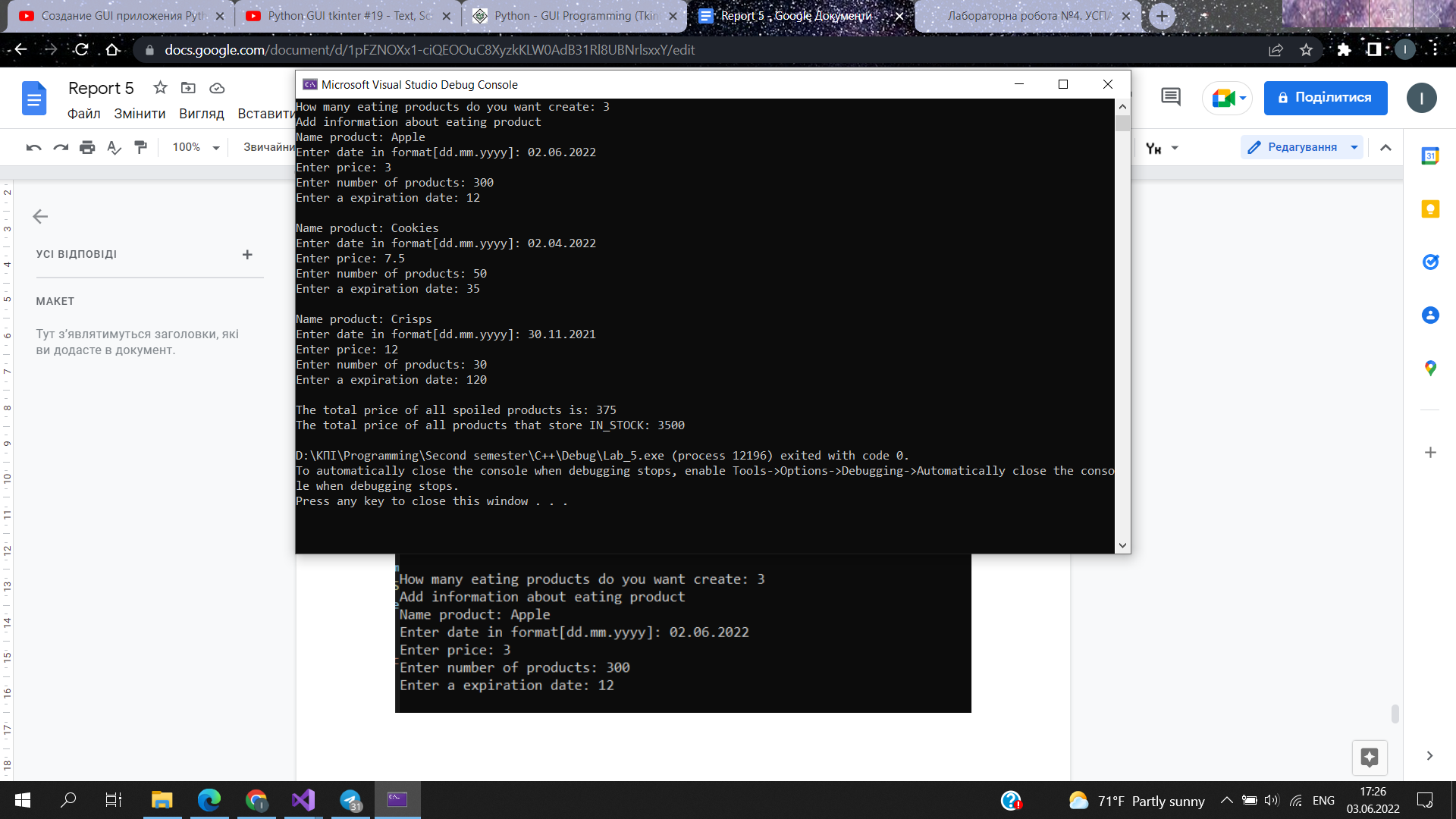
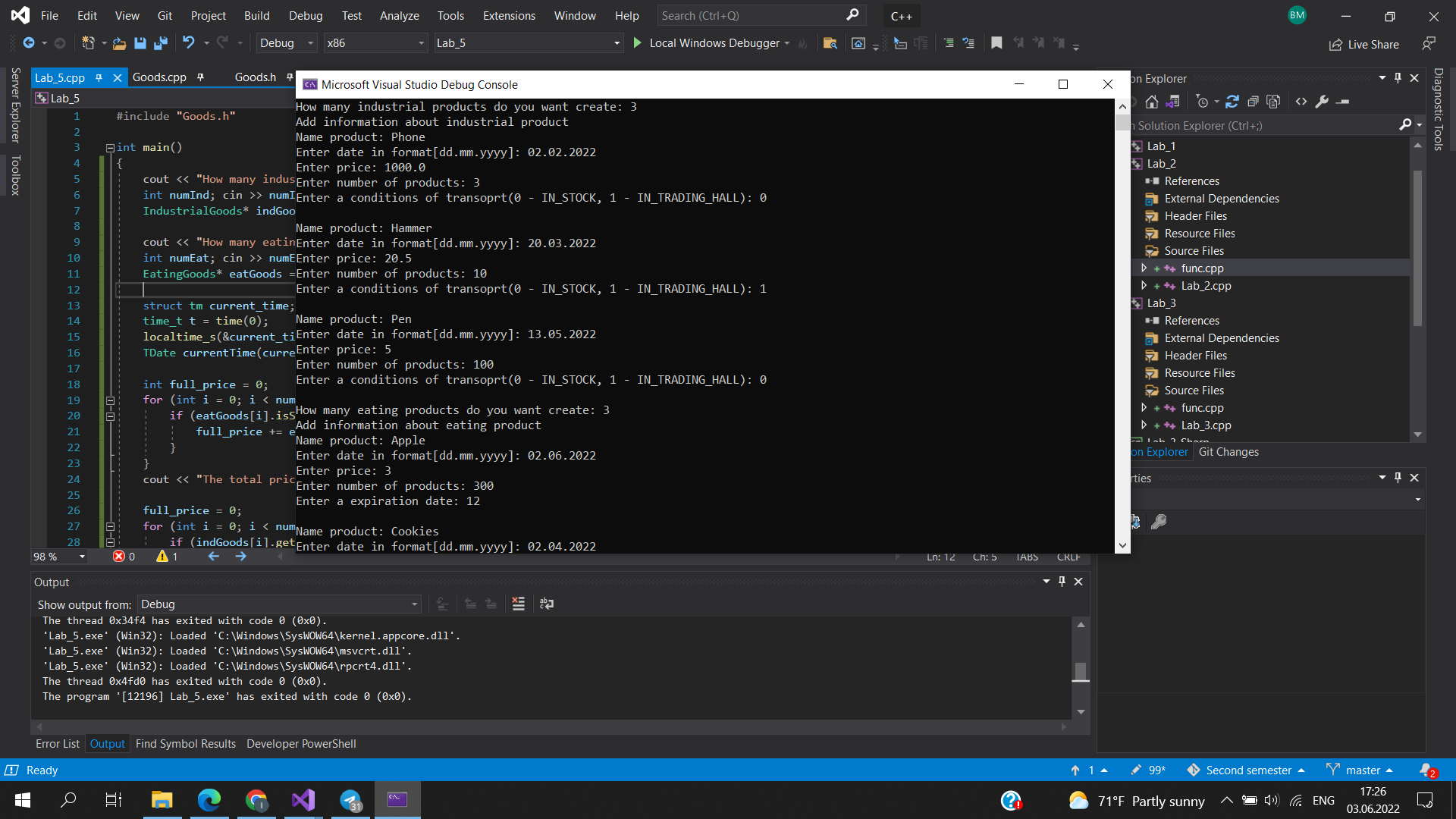
bool isSpoiledGood(TDate&);

};

IndustrialGoods\* createInd(int numInd);

EatingGoods\* createEat(int numEat);

**Тестування програми:**



*2. Виконання завдання на мові Python:*

**// Lab\_5.py**

from lib import \*

num\_ind\_goods = int(input("How many industrial products do you want create: "))

ind\_goods = create\_ind\_goods(num\_ind\_goods)

num\_eating\_goods = int(input("How many eating products do you want create: "))

eating\_goods = create\_eating\_goods(num\_eating\_goods)

time = datetime.now()

current\_time = TDate(time.day, time.month, time.year)

full\_price = 0

for i in range(num\_eating\_goods):

if eating\_goods[i].is\_spoiled\_good(current\_time):

full\_price += eating\_goods[i].get\_sum()

print("The total price of all spoiled products is:", full\_price)

full\_price = 0

for i in range(num\_ind\_goods):

if ind\_goods[i].get\_conditions() == "IN\_STOCK":

full\_price += ind\_goods[i].get\_sum()

print("The total price of all products that store IN\_STOCK:", full\_price)

**// lib.py**

from datetime import datetime

MONTH\_DATE = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

class TDate:

def \_\_init\_\_(self, day, month, year):

if year < 1900 or year > 2022:

print("Incorrect enter year")

exit(0)

if month <= 0 or month > 12:

print("Invalid enter month")

exit(0)

if day <= 0 or day > MONTH\_DATE[month - 1]:

print("Invalid entered day")

exit(0)

self.day = day

self.month = month

self.year = year

def add\_day(self, day):

self.day += day

while True:

if self.day > MONTH\_DATE[self.month - 1]:

self.day -= MONTH\_DATE[self.month - 1]

self.month += 1

else:

break

class Goods:

def \_\_init\_\_(self, name, date, price, number):

self.\_name = name

self.\_date = date

if price < 0:

print("Incorrect entered price")

exit(0)

if number <= 0:

print("Invalid number of products")

exit(0)

self.\_price = price

self.\_number = number

def get\_sum(self):

return self.\_price \* self.\_number

class Industrial\_Goods(Goods):

def \_\_init\_\_(self, name, date, price, number, conditions):

super().\_\_init\_\_(name, date, price, number)

self.\_\_conditions = conditions

def get\_conditions(self):

return self.\_\_conditions

class Eating\_Goods(Goods):

def \_\_init\_\_(self, name, date, price, number, expiration):

super().\_\_init\_\_(name, date, price, number)

if expiration < 0:

print("Incorrected expiration date")

exit(0)

self.\_\_expiration = expiration

def is\_spoiled\_good(self, current\_date):

expiration\_date = self.\_date

expiration\_date.add\_day(self.\_\_expiration)

current\_days = current\_date.year \* 365 + (current\_date.month - 1) \* 31 + current\_date.day

expiration\_days = expiration\_date.year \* 365 + (expiration\_date.month - 1) \* 31 + expiration\_date.day

if current\_days > expiration\_days:

return True

elif expiration\_days >= current\_days:

return False

def create\_ind\_goods(num\_ind\_goods):

ind\_goods = []

print("Add information about industrial product")

for i in range(num\_ind\_goods):

name = input("Name product: ")

date = input("Enter date in format[dd.mm.yyyy]: ")

date = date.split('.')

for i in range(3):

date[i] = int(date[i])

price = float(input("Enter price: "))

number = int(input("Enter number of products: "))

conditions = input("Enter a conditions of transoprt(IN\_STOCK, IN\_TRADING\_HALL): ")

ind\_goods.append(Industrial\_Goods(name, TDate(date[0], date[1], date[2]), price, number, conditions))

print()

return ind\_goods

def create\_eating\_goods(num\_eating\_goods):

eating\_goods = []

print("Add information about eating product")

for i in range(num\_eating\_goods):

name = input("Name product: ")

date = input("Enter date in format[dd.mm.yyyy]: ")

date = date.split('.')

for i in range(3):

date[i] = int(date[i])

price = float(input("Enter price: "))

number = int(input("Enter number of products: "))

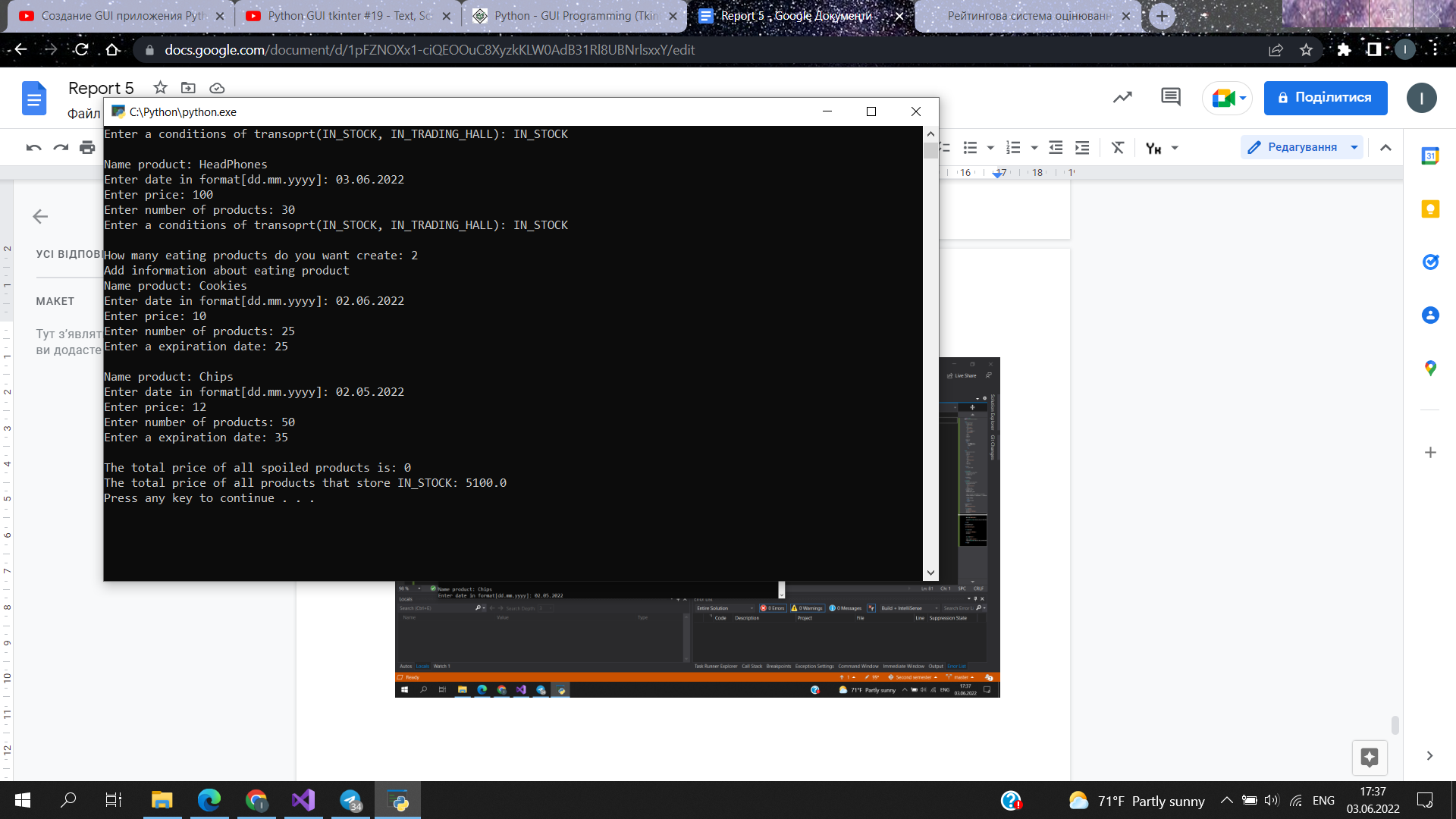
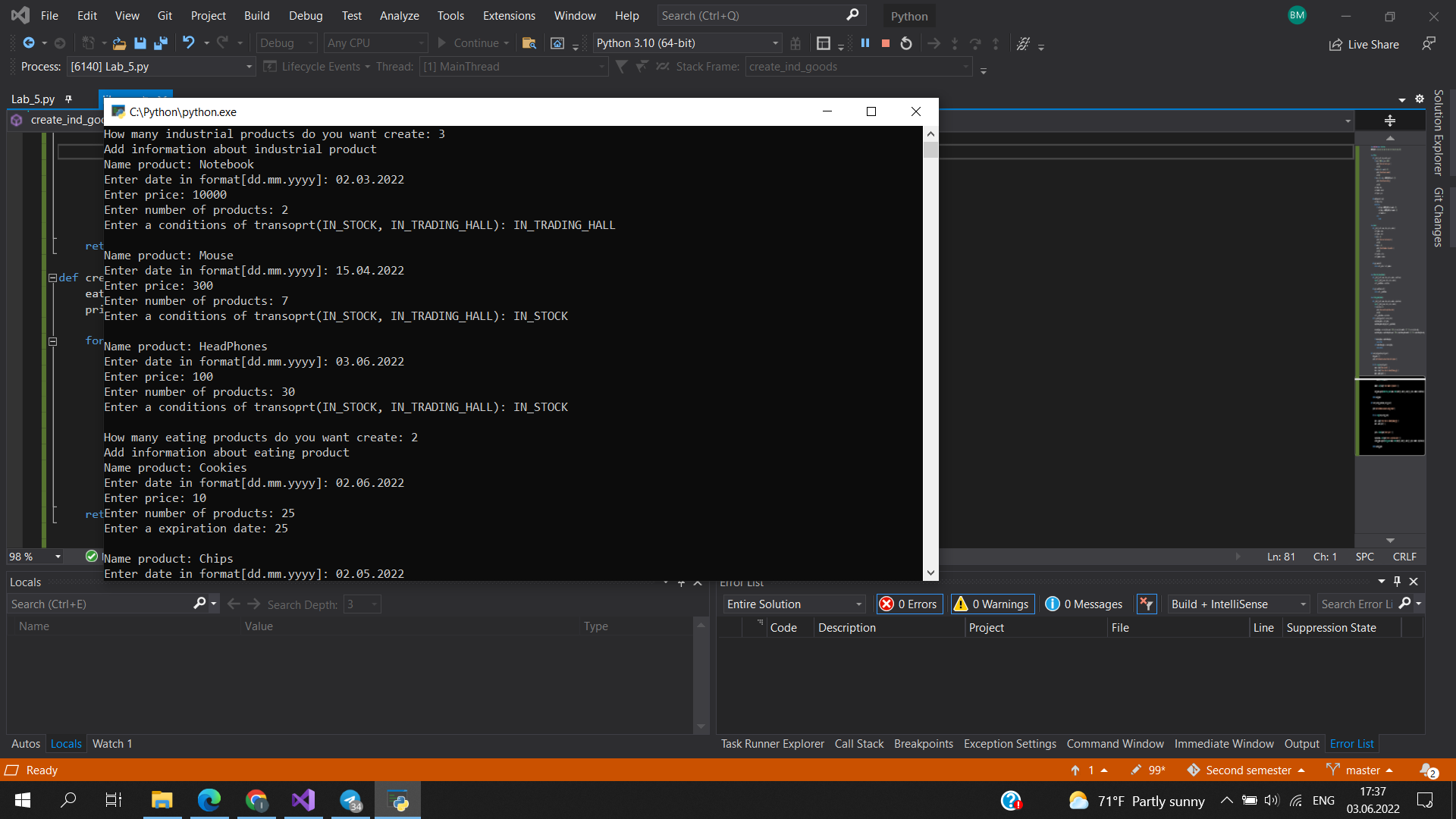
expiriation = int(input("Enter a expiration date: "))

eating\_goods.append(Eating\_Goods(name, TDate(date[0], date[1], date[2]), price, number, expiriation))

print()

return eating\_goods

**Тестування програми:**



**Висновок**: Під час виконання лабораторної роботи я вивчив особливості створення класів з використанням перевантажених операторів (операцій) на прикладі мови С++. Результатом виконання лабораторної роботи є програма, основним завданням якої є створення класу вектору та ініціалізація трьох його об’єктів. Після тестування програм можна зробити висновок, що вони справляються із поставленою задачею.