# Національний технічний університет України

*Додаток 1*

# «Київський політехнічний інститут імені Ігоря Сікорського»

**Навчально-науковий інститут атомної і теплової енергетики**

**Кафедра цифрових технологій в енергетиці**

**ЗВІТ**

# з лабораторної роботи № 4

**з дисципліни «Програмування на мові Java»**

# Тема 4. «Розробка програм в середовищі INTELLIJ IDEA для дослідження концепції математичного моделювання та об’єктно-орієнтованого програмування»

Варіант № 25

Виконав: Студент групи ТР-12

Руденко Владислав

Дата здачі: 29.10

Київ – 2023

**Завдання**

1. Написати програму мовою Java. Варіант обрати за списком групи. Представити виконання в Intellij IDEA. Продемонструвати детальні скріни виконання коду програми з поясненням. Обов’язково – наявність висновків.

2. Написати програму мовою Java. Варіант обрати за списком групи. Представити виконання в Intellij IDEA. Продемонструвати детальні скріни виконання коду програми з поясненням. Обов’язково – наявність висновків.

3. Написати програму мовою Java. Варіант обрати за списком групи. Представити виконання в Intellij IDEA. Продемонструвати детальні скріни виконання коду програми з поясненням. Обов’язково – наявність висновків.

4. Зробити звіт з лабораторної роботи та вчасно надіслати викладачу на перевірку (дедлайн для надсилання звітів по Лаб\_4 – до 27.11.2023 року до 23:59).

**ЗАВДАННЯ 1**

**Варіант 1 (для всіх студентів, хто в списку йде за непарним номером (1,3,5,7 і т д))**

Написати програму на Java, яка обчислює значення A та B за заданими формулами. Вимоги: -використати метод класу Math для обчислень ступенів, тригонометричних функцій тощо; -перевірити результати, шляхом виконання тестування з різними значеннями x та y для перевірки коректності розрахунків та виведення результатів

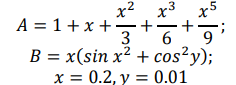


Рис. 1 Формули завдання

**ЗАВДАННЯ 2**

**Варіант 1 (для всіх студентів, хто в списку йде за непарним номером (1,3,5,7 і т д))**

Написати програму на Java, яка обчислює значення D та F за заданими формулами. Вимоги: -використати метод класу Math для обчислень ступенів, тригонометричних функцій тощо; -перевірити результати, шляхом виконання тестування з різними значеннями x, у та z для перевірки коректності розрахунків та виведення результатів.

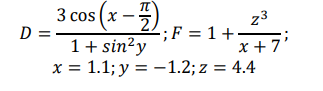


Рис. 2 Формули Завдання

**ЗАВДАННЯ 3**

**Варіант 1 (для всіх студентів, хто в списку йде за непарним номером (1,3,5,7 і т д))**

Написати програму на Java, в якій буде створено ієрархію класів для геометричних фігур. Наприклад, базовий клас «Фігура» і підкласи «Коло», «Прямокутник», «Трикутник». Кожен клас повинен включати методи для обчислення площі/периметру фігури. Використати наслідування для спрощення коду.

Рекомендації:

-клас «Фігура» може бути абстрактним, та може містити абстрактні методи: calculateArea()/calculatePerimeter().

-кожен підклас («Коло», «Прямокутник», «Трикутник») повинен реалізувати абстрактні методи calculateArea() та calculatePerimeter() згідно з їхніми геометричними формулами;

- кожен підклас повинен мати властивості, що визначають фігуру (наприклад, радіус для кола, довжину та ширину для прямокутника, сторони для трикутника). Рекомендації:

- клас «Транспорт» може бути абстрактним, та може містити атрибути, такі як швидкість, вага, колір тощо.

- може містити конструктор, геттери та сеттери для атрибутів; - кожен підклас («Автомобіль», «Велосипед», «Мотоцикл») повинен реалізувати абстрактні методи та можуть додавати додаткові атрибути та методи, специфічні для кожного типу транспортного засобу;

- для демонстрації створення об'єктів кожного з підкласів та виклику їхніх методів можна створити клас TransportTest;

- кожен підклас може містити конструктори для ініціалізації цих властивостей.

- для демонстрації створення об'єктів кожного з підкласів та виклику їхніх методів можна створити клас GeometryTest;

-необхідно додати перевірку вхідних даних у конструкторах підкласів, щоб запобігти створенню фігур з некоректними параметрами;

- обов’язково виведення на консоль результатів обчислень площі та периметру для кожної фігури.

**ЗАВДАННЯ 4\***

*(завдання додаткове, для бажаючих отримати додаткові бали (оцінюється в 3 бали). Вимоги щодо оформлення такі ж як і для інших завдань)*

Написати програму на Java, в якій буде створено три класи: «Book», «Library», та «Librarian». Кожен клас представляє окремий аспект системи управління бібліотекою. Клас «Book» представляє книги з назвою та автором, клас «Library» представляє бібліотеку зі списком книг, а клас «Librarian» представляє бібліотекаря, який відповідає за управління книгами. Код повинен представити структуру для моделювання бібліотеки. Основне завдання коду - додавання книг до бібліотеки, зберігання їх та відображення списку книг на консолі. Задача полягає в створенні програми для керування книгами у бібліотеці та відображенні інформації про них.

Рекомендації:

1)Клас «Book»: -повинен містити принаймні два атрибути: title (назва) та author (автор); -потрібно надати геттери та сеттери для управління зазначеними атрибутами; - можна реалізувати метод toString() для виведення інформації про книги.

2)Клас «Library»: - повинен містити колекцію книг, наприклад, використовуючи ArrayList. - методи можуть включати: addBook, removeBook, findBookByTitle, listAllBooks, тощо. - можна реалізувати додатково пошук книг за автором або жанром.

3)Клас «Librarian»: - може включати методи для взаємодії з «Library», наприклад, manageBookAddition(), manageBookRemoval(). - може містити методи для взаємодії з користувачами, такі як checkOutBook()/returnBook().

При виконанні завдання, студент повинен використати підготовлений список з 5-7 реальних книг з назвами та авторами для заповнення бібліотеки.

При демонстрації взаємодії класів, студент повинен підготувати приклад, який продемонструє, як об'єкти класів можуть взаємодіяти між собою, наприклад, як Librarian викликає методи Library для додавання або видалення книг

**Хід виконання та Результати**

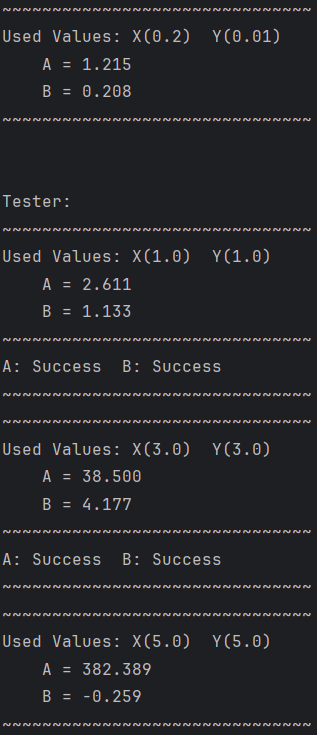


Рис. 3 Результат виконання Завдання 1

**Висновок:** Під час виконання лабораторної роботи було розроблено програму на мові програмування Java для обчислення значень змінних A та B за допомогою заданих формул. Для досягнення цієї мети вимагалося використання методів класу Math для обчислення ступенів, тригонометричних функцій та інших математичних операцій.

В рамках виконання завдання було проведено тестування програми з різними значеннями x та y з метою перевірки коректності розрахунків і правильності виведення результатів. Це дозволило переконатися у відповідності отриманих результатів очікуваним значенням згідно заданих формул.

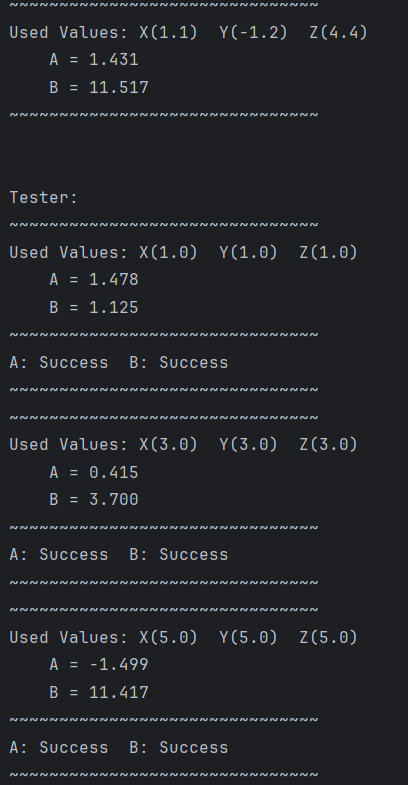


Рис. 4 Результат виконання Завдання 2

**Висновок:** Під час виконання завдання було розроблено програму на мові програмування Java для обчислення значень змінних D та F згідно з заданими формулами. Для виконання цього завдання було використано методи класу Math для обчислення ступенів, тригонометричних функцій та інших математичних операцій.

З метою перевірки правильності обчислень та коректності результатів було проведено тестування програми з різними значеннями x, у та z. Це дозволило переконатися, що отримані результати відповідають очікуваним значенням відповідно до заданих формул.

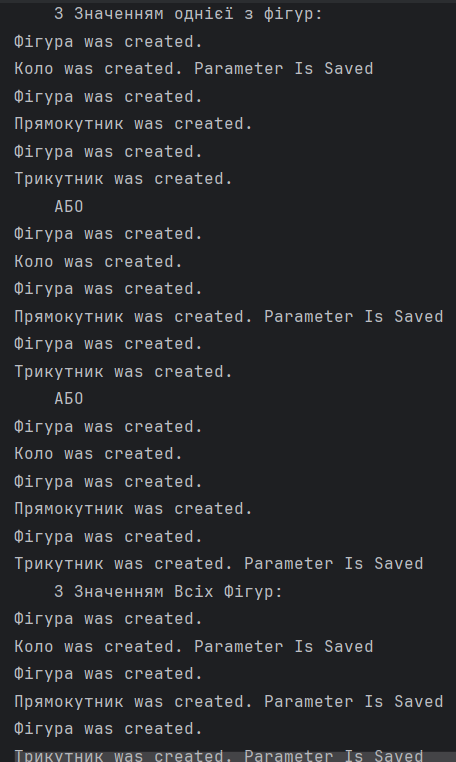


Рис. 5 Результат виконання Завдання 3 (Демонстрація реалізації наслідування з викликом конструкторів)

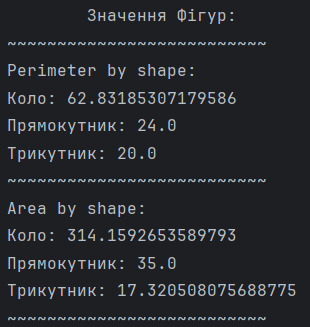


Рис. 6 Результат виконання Завдання 3 (Обчислення значень Периметру та Площі згідно до заданих параметрів)

**Висновок:** Під час роботи над завданням було створено програму на мові програмування Java, де була реалізована ієрархія класів для різних геометричних фігур. Згідно з вказаним завданням, був створений базовий клас "Фігура" і підкласи "Коло", "Прямокутник" і "Трикутник".

У кожному класі були реалізовані методи для обчислення площі та периметру відповідної фігури. Для спрощення коду та використання спадковості було застосовано наслідування: класи-підкласи успадковували методи базового класу "Фігура", що дозволило уникнути дублювання коду та забезпечило зручність у роботі з різними типами фігур.

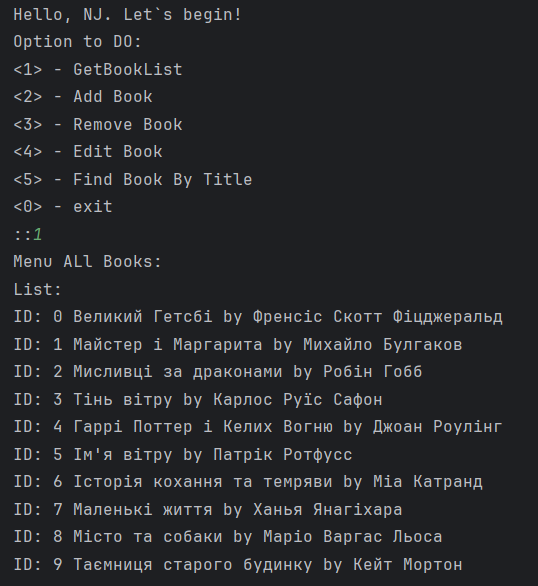


Рис. 7 Виконання Додаткового Завдання 4 (Демонстрація виводу всіх існуючих книг)

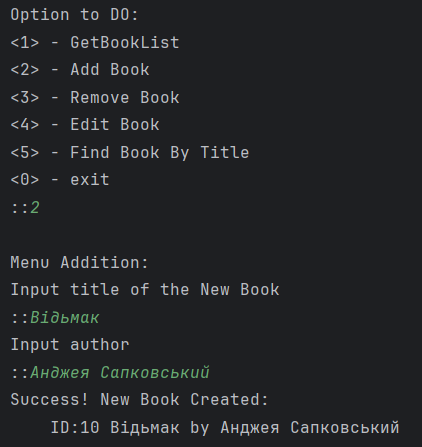


Рис. 8 Виконання Додаткового Завдання 4 (Демонстрація Додавання нової книги до білбіотеки)

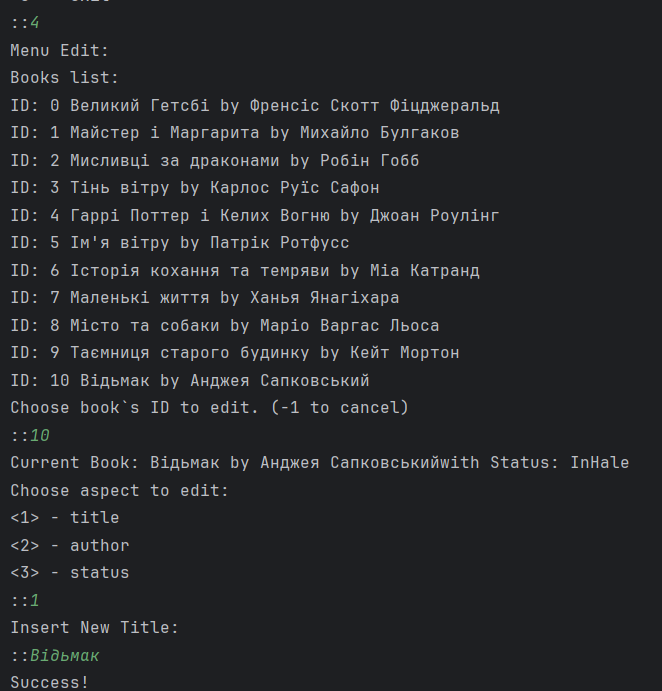


Рис. 9 Виконання Додаткового Завдання 4 (Редагування Існуючої Книги)

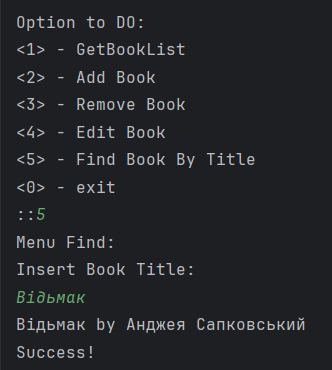


Рис. 10 Виконання Додаткового Завдання 4 (Демонстрація пошуку за Назвою)

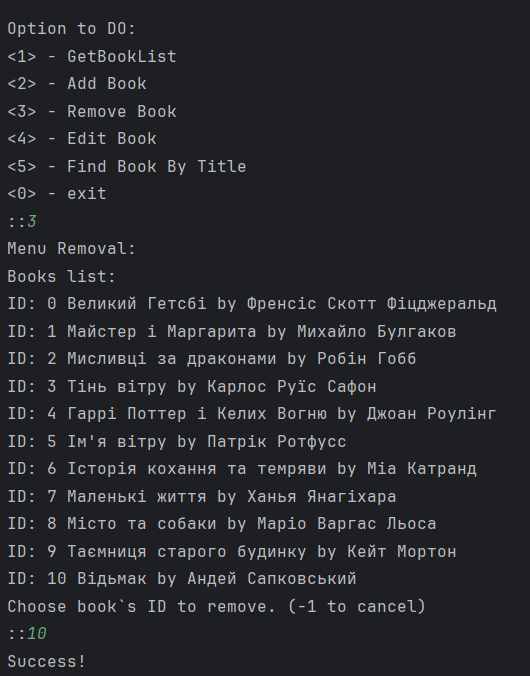


Рис. 11 Виконання Додаткового Завдання 4 (Демонстрація Видалення книги з бібліотеки)

**Висновок:** Під час розробки програми було створено три класи: "Book", "Library" та "Librarian", що відображають аспекти управління бібліотекою. Клас "Book" представляє книги з вказаною назвою та автором, "Library" представляє бібліотеку зі списком книг, а "Librarian" відповідає за керування цими книгами.Під час виконання завдання було використано список з 5-7 реальних книг з вказаними назвами та авторами для заповнення бібліотеки. Програма була розроблена з урахуванням вказаних вимог та взаємодіяла з об'єктами класів у відповідності зі сформованою логікою взаємодії між класами.

*Додаток 1*

**Лістинг**

**Завдання 1**

package Lab4;

import java.math.BigDecimal;

import java.math.RoundingMode;

import static java.lang.Math.\*;

public class Lab4Ex1 {

public static void main(String[] args)

{

double X = 0.2;

double Y = 0.01;

Solver(X,Y);

Tester();

}

static void Tester()

{

double[][] Values = new double[][]{

{1,1,2.611,1.133},

{3,3,38.5,4.177},

{5,5,382.389,-0.259},

{7,9,1948.944,-0.865}

};

System.out.println("\n\nTester:");

for (double[] value : Values) {

double[] result = Solver(value[0], value[1]);

System.out.print("A: " + (result[0] == value[2] ? "Success" : "Fail"));

System.out.println("\tB: " + (result[1] == value[3] ? "Success" : "Fail"));

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

}

}

static double[] Solver(double X, double Y)

{

BigDecimal A = BigDecimal.valueOf(AFunction(X)).setScale(3, RoundingMode.HALF\_UP);

BigDecimal B = BigDecimal.valueOf(BFunction(X, Y)).setScale(3, RoundingMode.HALF\_UP);

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("Used Values: X("+X+") Y("+Y+")");

System.out.println("\tA = "+A+"\n\tB = "+B);

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

return new double[]{A.doubleValue(),B.doubleValue()};

}

static double AFunction(double X){

return 1+X+(pow(X,2)/3)+(pow(X,3)/6)+(pow(X,5)/9);

}

static double BFunction(double X, double Y){

return X\*(sin(pow(X,2))+pow(cos(Y),2));

}

}

**Завдання 2**

package Lab4;

import java.math.BigDecimal;

import java.math.RoundingMode;

import static java.lang.Math.\*;

public class Lab4Ex2 {

public static void main(String[] args)

{

double X = 1.1;

double Y = -1.2;

double Z = 4.4;

Solver(X,Y,Z);

Tester();

}

static void Tester()

{

double[][] Values = new double[][]{

//X,Y,Z,FResult,SResult

{1,1,1,1.478,1.125},

{3,3,3,0.415,3.700},

{5,5,5,-1.499,11.417},

{7,9,11,1.685,96.071}

};

System.out.println("\n\nTester:");

for (double[] value : Values) {

double[] result = Solver(value[0], value[1],value[2]);

System.out.print("A: " + (result[0] == value[3] ? "Success" : "Fail"));

System.out.println("\tB: " + (result[1] == value[4] ? "Success" : "Fail"));

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

}

}

static double[] Solver(double X, double Y,double Z)

{

BigDecimal D = BigDecimal.valueOf(DFunction(X,Y)).setScale(3, RoundingMode.HALF\_UP);

BigDecimal F = BigDecimal.valueOf(FFunction(X,Z)).setScale(3, RoundingMode.HALF\_UP);

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("Used Values: X("+X+") Y("+Y+") Z("+Z+")");

System.out.println("\tA = "+D+"\n\tB = "+F);

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");

return new double[]{D.doubleValue(),F.doubleValue()};

}

static double DFunction(double X,double Y){

return (3\*cos(X-PI/2))/(1+pow(sin(Y),2));

}

static double FFunction(double X, double Z){

return 1+(pow(Z,3))/(X+7);

}

}

**Завдання 3**

package Lab4;

import static java.lang.Math.\*;

public class Lab4Ex3 {

public static void main(String[] args)

{

System.out.println("\t\tКонструктори:");

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("\tБез Значень Фігур:");

GeometryTest geo = new GeometryTest();

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("\tЗ Значенням однієї з фігур:");

geo = new GeometryTest(10);

System.out.println("\tАБО");

geo = new GeometryTest(5,7);

System.out.println("\tАБО");

geo = new GeometryTest(8,7,5);

System.out.println("\tЗ Значенням Всіх Фігур:");

geo = new GeometryTest(10,5,7,8,7,5);

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("\n\n\t\tЗначення Фігур:");

geo.CallAllShapes();

}

}

class GeometryTest

{

Коло circle;

Прямокутник cube;

Трикутник triangle;

public GeometryTest()

{

this.circle=new Коло();

this.cube=new Прямокутник();

this.triangle=new Трикутник();

}

public GeometryTest(double R)

{

this.circle=new Коло(R);

this.cube=new Прямокутник();

this.triangle=new Трикутник();

}

public GeometryTest(double height,double width)

{

this.circle=new Коло();

this.cube=new Прямокутник(height,width);

this.triangle=new Трикутник();

}

public GeometryTest(double Aside, double Bside, double Cside)

{

this.circle=new Коло();

this.cube=new Прямокутник();

this.triangle=new Трикутник(Aside,Bside,Cside);

}

public GeometryTest(double R,double height,double width, double Aside, double Bside, double Cside)

{

this.circle=new Коло(R);

this.cube=new Прямокутник(height,width);

this.triangle=new Трикутник(Aside,Bside,Cside);

}

public void CallAllShapes()

{

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("Perimeter by shape:");

System.out.println("Коло: "+this.circle.calculatePerimeter());

System.out.println("Прямокутник: "+this.cube.calculatePerimeter());

System.out.println("Трикутник: "+this.triangle.calculatePerimeter());

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

System.out.println("Area by shape:");

System.out.println("Коло: "+this.circle.calculateArea());

System.out.println("Прямокутник: "+this.cube.calculateArea());

System.out.println("Трикутник: "+this.triangle.calculateArea());

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~");

}

}

abstract class Фігура

{

protected boolean Enable = true;

public abstract double calculateArea();

public abstract double calculatePerimeter();

public Фігура()

{

System.out.println("Фігура was created.");

}

public boolean isEnable() {

return Enable;

}

}

class Коло extends Фігура

{

private double R=0;

public Коло()

{

System.out.println("Коло was created.");

Enable = false;

}

public Коло(double R)

{

if(R<=0)

{

System.out.println("Error! Incorrect Value");

Enable = false;

return;

}

System.out.println("Коло was created. Parameter Is Saved");

this.R = R;

}

@Override

public double calculateArea() {

return Enable?PI\*pow(R,2):0;

}

@Override

public double calculatePerimeter() {

return Enable?2\*PI\*R:0;

}

public double getR() {

return R;

}

}

class Прямокутник extends Фігура

{

private double height=0, width=0;

public Прямокутник()

{

System.out.println("Прямокутник was created.");

Enable = false;

}

public Прямокутник(double height, double width)

{

if(height<=0 || width<=0)

{

System.out.println("Error! Incorrect Value");

Enable = false;

return;

}

System.out.println("Прямокутник was created. Parameter Is Saved");

this.height = height;

this.width = width;

}

@Override

public double calculateArea() {

return Enable?height\*width:0;

}

@Override

public double calculatePerimeter() {

return Enable?2\*(height+width):0;

}

public double getWidth() {

return width;

}

public double getHeight() {

return height;

}

}

class Трикутник extends Фігура

{

private double Aside=0, Bside=0,Cside=0;

public Трикутник()

{

System.out.println("Трикутник was created.");

}

public Трикутник(double Aside, double Bside, double Cside)

{

if(Aside<=0 || Bside<=0 || Aside + Bside <= Cside || Aside + Cside <= Bside || Bside + Cside <= Aside)

{

System.out.println("Error! Incorrect Value");

Enable = false;

return;

}

System.out.println("Трикутник was created. Parameter Is Saved");

this.Aside = Aside;

this.Bside = Bside;

this.Cside = Cside;

}

@Override

public double calculateArea() {

double S = (Aside+Bside+Cside)/2;

return Enable?Math.sqrt(S \* (S - Aside) \* (S - Bside) \* (S - Cside)):0;

}

@Override

public double calculatePerimeter() {

return Enable?Aside+Bside+Cside:0;

}

public double getAside() {

return Aside;

}

public double getBside() {

return Bside;

}

public double getCside() {

return Cside;

}

}

**Завдання 4**

package Lab4;

import java.util.ArrayList;

import java.util.Objects;

import java.util.Scanner;

public class Lab4Ex4 {

public static void main(String[] args)

{

Library lib = new Library();

LibraryBoot boot = new LibraryBoot(lib);

Librarian libworker = new Librarian("NJ",lib);

}

}

class LibraryBoot

{

Library lib;

String[][] testtext = new String[][]{

{"Великий Гетсбі", "Френсіс Скотт Фіцджеральд"},

{"Майстер і Маргарита", "Михайло Булгаков"},

{"Мисливці за драконами", "Робін Гобб"},

{"Тінь вітру", "Карлос Руїс Сафон"},

{"Гаррі Поттер і Келих Вогню", "Джоан Роулінг"},

{"Ім'я вітру", "Патрік Ротфусс"},

{"Історія кохання та темряви", "Міа Катранд"},

{"Маленькі життя", "Ханья Янагіхара"},

{"Місто та собаки", "Маріо Варгас Льоса"},

{"Таємниця старого будинку", "Кейт Мортон"}

};

public LibraryBoot(Library lib)

{

for(int i=0;i<testtext.length;i++)

{

lib.addBook(testtext[i][0],testtext[i][1]);

}

}

}

enum BookStatus

{

Borrowed,

InHale

}

class Book

{

private String title;

private String author;

private BookStatus status;

Book(String title, String author)

{

this.author=author;

this.title=title;

status=BookStatus.InHale;

}

public String toString()

{

return title+" by "+author;

}

public boolean titleMatch(String title)

{

return (Objects.equals(this.title, title));

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public BookStatus getStatus() {

return status;

}

public void setStatus(BookStatus status) {

this.status = status;

}

}

class Library{

private ArrayList<Book> Books = new ArrayList<>();

public void addBook(String title, String author)

{

Books.add(new Book(title, author));

}

public void removeBook(int ID)

{

Books.remove(ID);

}

public int lastID()

{

return Books.size()-1;

}

public void changeBook(int ID, int Key, String Value)

{

Book book = Books.get(ID);

switch (Key)

{

case 1:

book.setTitle(Value);

break;

case 2:

book.setAuthor(Value);

break;

}

}

public void changeBook(int ID, int Key, BookStatus status)

{

Book book = Books.get(ID);

book.setStatus(status);

}

public int findBookByTitle(String title)

{

int i;

boolean found = false;

for(i=0;i<Books.size();i++)

{

if(Books.get(i).titleMatch(title))

{

found = true;

break;

}

}

return found?i:-1;

}

public void listAllBooks()

{

int i=0;

for(Book book:Books)

{

System.out.println("ID: "+i+" "+ book.toString());

i++;

}

}

public Book getBook(int ID)

{

return Books.get(ID);

}

}

class Librarian{

private Library currentlibrary;

private String Name;

private Scanner scan = new Scanner(System.in);

Librarian(String Name,Library library)

{

this.currentlibrary=library;

this.Name=Name;

Manager();

}

private void Manager()

{

int choose = 0;

System.out.println("Hello, "+Name+". Let`s begin!");

do

{

System.out.print("Option to DO:\n<1> - GetBookList\n<2> - Add Book\n<3> - Remove Book\n<4> - Edit Book\n<5> - Find Book By Title\n<0> - exit\n::");

choose = scan.nextInt();

switch (choose)

{

case 0:

choose = -1;

break;

case 1:

bookList();

break;

case 2:

manageBookAddition();

break;

case 3:

manageBookRemoval();

break;

case 4:

manageBookEdit();

break;

case 5:

findBook();

break;

case -1:

System.out.println("\t\tExiting...");

break;

default:

System.out.println("InCorrect! Try Again.");

break;

}

}while(choose != -1);

}

public void manageBookAddition()

{

scan.nextLine();

System.out.print("\nMenu Addition:\nInput title of the New Book\n::");

String title = scan.nextLine();

System.out.print("Input author\n::");

String author = scan.nextLine();

currentlibrary.addBook(title,author);

System.out.println("Success! New Book Created:\n\t"+"ID:"+currentlibrary.lastID()+" "+currentlibrary.getBook(currentlibrary.lastID()).toString());

System.out.println();

}

public void manageBookRemoval()

{

scan.nextLine();

System.out.println("Menu Removal:\nBooks list:");

currentlibrary.listAllBooks();

System.out.print("Choose book`s ID to remove. (-1 to cancel)\n::");

int choose = scan.nextInt();

if(choose == -1)

{

System.out.println("Exiting...");

return;

}

currentlibrary.removeBook(choose);

System.out.println("Success!\n");

}

public void manageBookEdit()

{

scan.nextLine();

System.out.println("Menu Edit:\nBooks list:");

currentlibrary.listAllBooks();

System.out.print("Choose book`s ID to edit. (-1 to cancel)\n::");

int chooseID = scan.nextInt();

if(chooseID == -1)

{

System.out.println("Exiting...");

return;

}

System.out.println("Current Book: "+currentlibrary.getBook(chooseID) + "with Status: "+currentlibrary.getBook(chooseID).getStatus().name());

System.out.print("Choose aspect to edit:\n<1> - title\n<2> - author\n<3> - status\n::");

int choose = scan.nextInt();

switch (choose)

{

case 1:

scan.nextLine();

System.out.print("Insert New Title:\n::");

String temp = scan.nextLine();

currentlibrary.changeBook(chooseID,1,temp);

break;

case 2:

scan.nextLine();

System.out.print("Insert New Author:\n::");

temp = scan.nextLine();

currentlibrary.changeBook(chooseID,2,temp);

break;

case 3:

System.out.print("Chosse new Status:\n<1> - "+BookStatus.Borrowed.name()+"\n<2> - "+BookStatus.InHale.name()+"\n::");

if(scan.nextInt() == 1)

{

currentlibrary.changeBook(chooseID,2,BookStatus.Borrowed);

}

else

{

currentlibrary.changeBook(chooseID,2,BookStatus.InHale);

}

break;

}

System.out.println("Success!\n");

}

public void findBook()

{

scan.nextLine();

System.out.println("Menu Find:\nInsert Book Title:");

String title = scan.nextLine();

if(currentlibrary.findBookByTitle(title) != -1)

{

System.out.println(currentlibrary.getBook(currentlibrary.findBookByTitle(title)).toString());

System.out.println("Success!");

}

else

{

System.out.println("Book Not Found!");

}

System.out.println();

}

public void bookList()

{

System.out.println("Menu ALl Books:\nList:");

if (currentlibrary.lastID() != -1) {

currentlibrary.listAllBooks();

} else {

System.out.println("List is clear");

}

System.out.println();

}

}