Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Факультет комп’ютерних наук

**Пояснювальна записка**

до курсової роботи з дисципліни «Об’єктно-орієнтоване програмування»

на тему: Бібліотека та управління реєстром видань в ній

Виконав студент 2 курсу

групи КС-21

Клочко Андрій Володимирович

Перевірив:

к.т.н., доц. Нагорний К.А.

Харків – 2020

**РЕФЕРАТ**

Звіт про виконання курсової роботи: 25 сторінок., 14 рисунків.

Ключові слова: ООП, JAVA, SOLID, IDE, CSV, ПРЕДМЕТНА ОБЛАСТЬ.

Пропонується розробити java-додаток для автоматизації процесу управління реєстром видань в бібліотеці. Спочатку треба визначити предметну область, проаналізувати вимоги, створити модель програмного продукту, написати програмну реалізацію.

Даний застосунок дозволить керувати реєстром публікацій в бібліотеці, а також реєстром користувачів за допомогою CLI. Модель програмного продукту буде побудована за допомогою UML-діаграми класів.

ЗМІСТ

[СПИСОК ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ 4](#_Toc59293263)

[ВСТУП 5](#_Toc59293264)

[РОЗДІЛ 1 ПОСТАНОВА ЗАДАЧІ 6](#_Toc59293265)

[1.1 Область визначення прикладного рішення 6](#_Toc59293266)

[1.2 Основні можливості системи 6](#_Toc59293267)

[РОЗДІЛ 2 ПРОЕКТУВАННЯ 8](#_Toc59293268)

[2.1 Опис предметної області 8](#_Toc59293269)

[2.2 Класи предметної області 8](#_Toc59293270)

[2.3 UML-діаграма класів 9](#_Toc59293271)

[РОЗДІЛ 3 РЕЗУЛЬТАТИ РОБОТИ 13](#_Toc59293272)

[3.1 Тестування 13](#_Toc59293273)

[3.2 Інструкція для користувача 16](#_Toc59293274)

[ВИСНОВОК 18](#_Toc59293275)

[ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 19](#_Toc59293276)

[ДОДАТОК А. ВИХІДНИЙ КОД 20](#_Toc59293277)

[Лістинг А.1 – клас Library 20](#_Toc59293278)

[Лістинг А.2 – клас PublicationRegister 21](#_Toc59293279)

[Лістинг А.3 – абстрактний клас Publication 22](#_Toc59293280)

[Лістинг А.4 – клас Book 23](#_Toc59293281)

[Лістинг А.5 – клас Magazine 24](#_Toc59293282)

[Лістинг А.6 – клас UserRegister 25](#_Toc59293283)

[Лістинг А.7 – клас User 26](#_Toc59293284)

# СПИСОК ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ООП – об’єктно-орієнтовне програмування.

ПрО (предметна область) – частина реального світу, що розглядається в межах певного контексту.

UML (англ. Unified Modeling Language) – уніфікована мова моделювання.

JDK (англ. Java Development Kit) – безкоштовний розповсюджуваний Oracle комплект розробника застосунків на мові Java.

SOLID – абревіатура п'яти основних принципів проектування в об'єктно-орієнтованому програмуванні – Single responsibility, Open-closed, Liskov substitution, Interface segregation і Dependency inversion. У перекладі на українську: принципи єдиної відповідальності, відкритості / закритості, підстановки Барбари Лісков, поділу інтерфейсу і інверсії залежностей).

IDE (Integrated Development Environment) – система програмних засобів, яка використовується програмістами для розробки програмного забезпечення.

CSV – текстовий формат, призначений для представлення табличних даних. Рядок таблиці відповідає рядку тексту, яка містить одне або кілька полів, розділених комами.

# ВСТУП

У ході роботи буде розроблений застосунок, який дозволить користувачам бібліотеки взаємодіяти із реєстром видінь, що зберігаються у бібліотеці.

При проектуванні програмного забезпечення, з використанням парадигми ООП, принципів проектування SOLID, моделі будуть побудовані за допомогою діаграм мови UML. Для побудування UML-діаграм класів та Use Case-діаграм буде використовуватись онлайн ресурс creately [1].

У ході розробки буде використовуватись мова програмування java, а також IDE, а саме Intellij IDEA від компанії JetBrains [2].

# РОЗДІЛ 1 ПОСТАНОВА ЗАДАЧІ

## 1.1 Область визначення прикладного рішення

**Мета**: Розробка прототипу програмної системи на базі основних принципів об’єктно-орієнтованого дизайну та сучасних практик програмування із використанням об’єктно-орієнтованої мови програмування Java.

**Предметна область:** бібліотека.

**Призначення програмної системи:** управління реєстром видань у бібліотеці.

**Задання**: Розробити прототип програмної системи (ПС) управління виданнями у бібліотеці, яка надає можливість користувачу маніпулювати бібліотечним реєстром. Інтерфейс користувача – консоль, до якої має бути виведений список усіх можливих команд. ПС має працювати до введення команди виходу із системи. Потрібно реалізувати збереження даних таких як публікації та користувачі, в файл у форматі CSV (coma separated values). Описати відповідну структуру класів розробленого програмного рішення в нотації UML у термінах діаграми класів (class diagram).

## 1.2 Основні можливості системи

До переліку можливостей системи входять такі обов’язкові пункти:

1. Додати видання до реєстру
2. Видалити видання з реєстру
3. Отримати список усіх видань, що містяться у реєстрі
4. Отримати список усіх видань за мовою якою вони надруковані
5. Видати обране видання користувачеві бібліотеки, при цьому видання не може бути видалене із реєстру до моменту повернення до бібліотеки
6. Зберегти реєстр до сховища (текстовий файл формату CSV)
7. Отримати реєстр зі сховища (текстового файлу формату CSV)

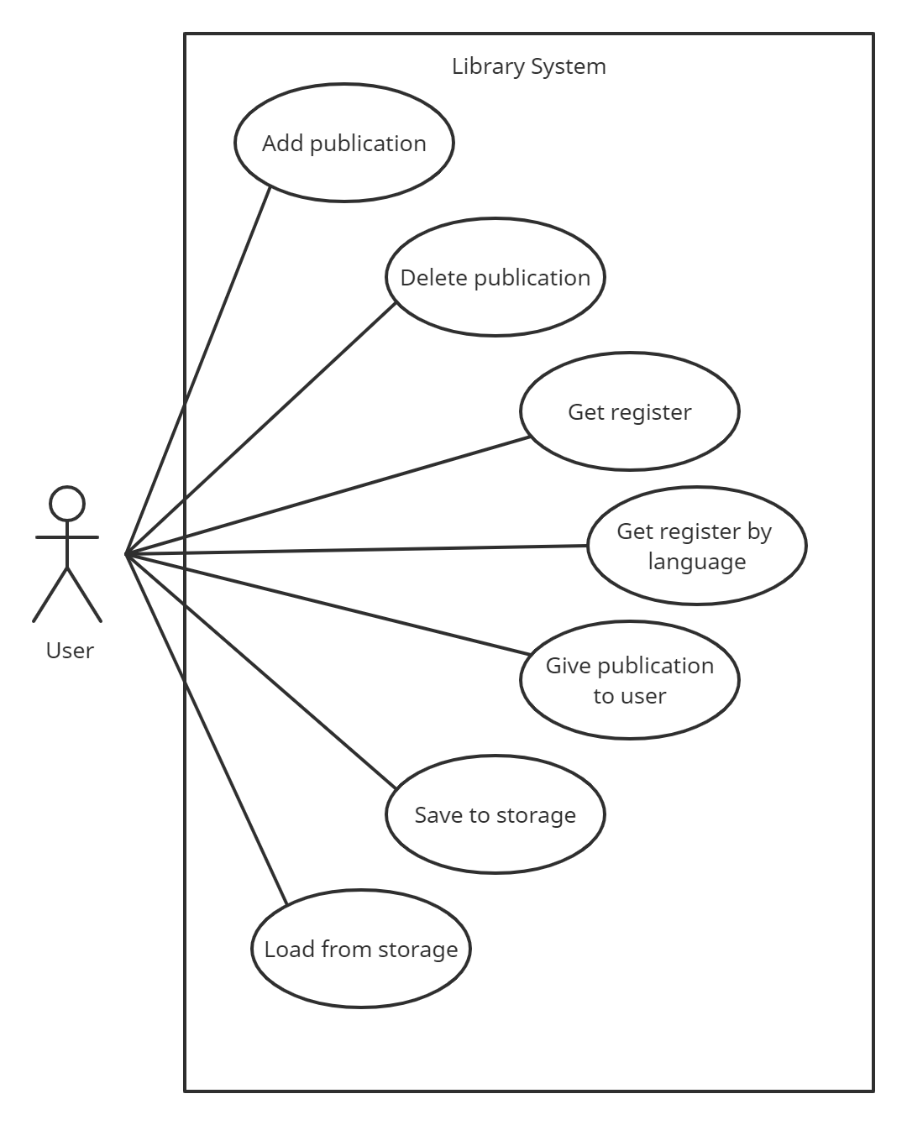


Рисунок 1.1 – Use Case-діаграма

# РОЗДІЛ 2 ПРОЕКТУВАННЯ

## 2.1 Опис предметної області

До основних сутностей предметної області (ПрО) входять такі сутності:

1. Бібліотека. Має назву, адресу, мастить реєстр видань
2. Книга – неперіодичне видання, яке має назву, автора, кількість сторінок та мову друку. У книги є видавець, який видає книгу у відповідному році.
3. Журнал – періодичне видання, яке має назву, автора статті, кількість статей та загальну кількість сторінок та мову друку. У журналу є видавець, який видає журнал у відповідному році.
4. Автор – людина, яка створила книгу або приймала участь у створенні журналу.
5. Користувач бібліотеки – має ім’я та прізвище, рік народження та номер бібліотечного квитка

## 2.2 Класи предметної області

Library – головний клас, має в собі реєстр публікацій та користувачів.

Publication – абстрактний клас, предок класів Book та Magazine.

Book, Magazine – різновид публікації, являються нащадками абстрактного класу Publication.

PublicationType – клас перерахувань типів публікацій.

PublicationRegister – реєстр публікацій, список всіх публікацій які знаходяться або знаходились в бібліотеці.

User – клас користувача системи.

UserRegister – реєстр користувачів, список всіх користувачів бібліотеки.

FileWriter – клас запису даних в файл.

OutputDataFormer – клас формування даних для запису у вихідний файл.

FileReader – клас зчитування даних з файлу.

InputDataParser – клас парсер даних з вхідного файлу.

## 2.3 UML-діаграма класів

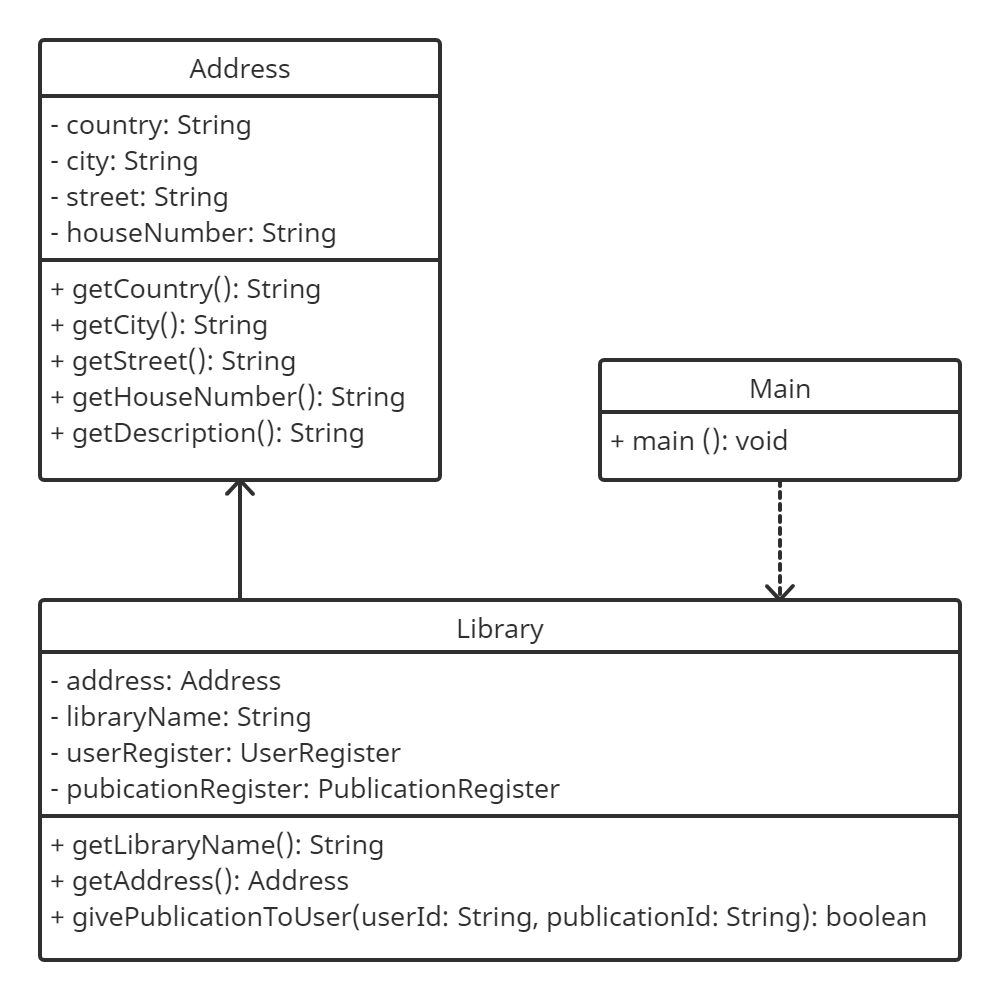


Рисунок 2.1 – UML-діаграма класу Library

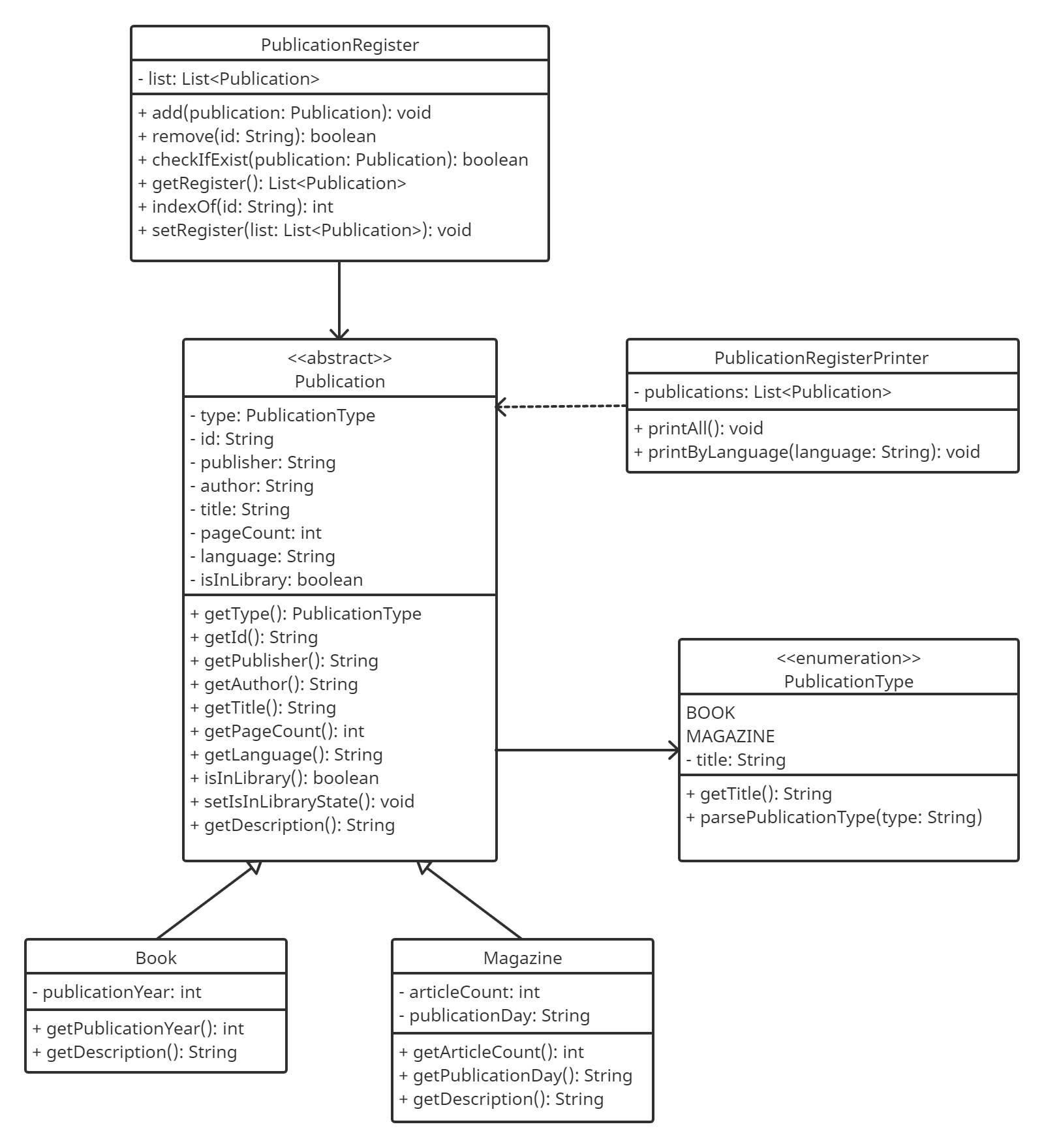


Рисунок 2.2 – UML-діаграма класу PublicationRegister

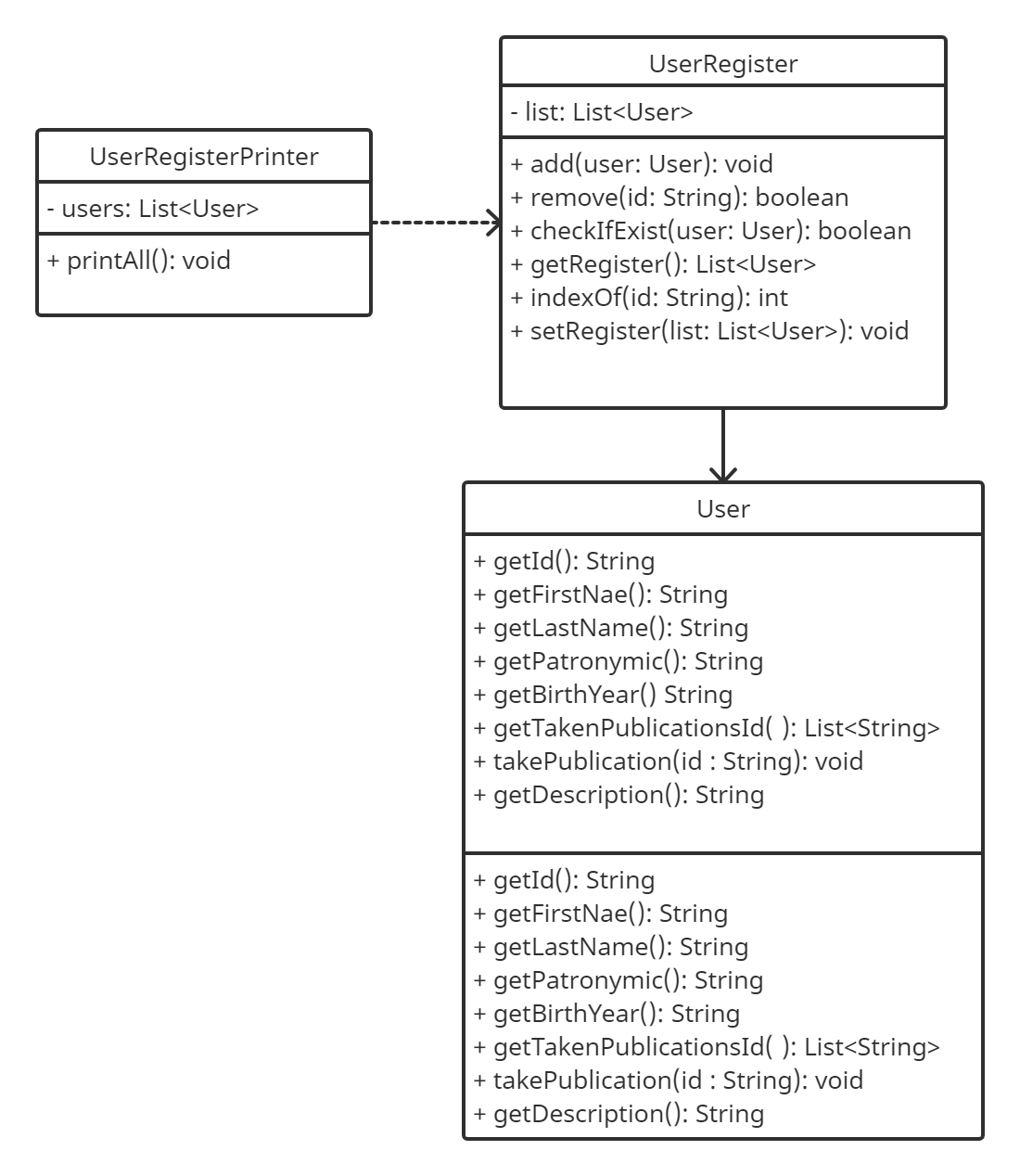


Рисунок 2.3 – UML-діаграма класу UserRegister

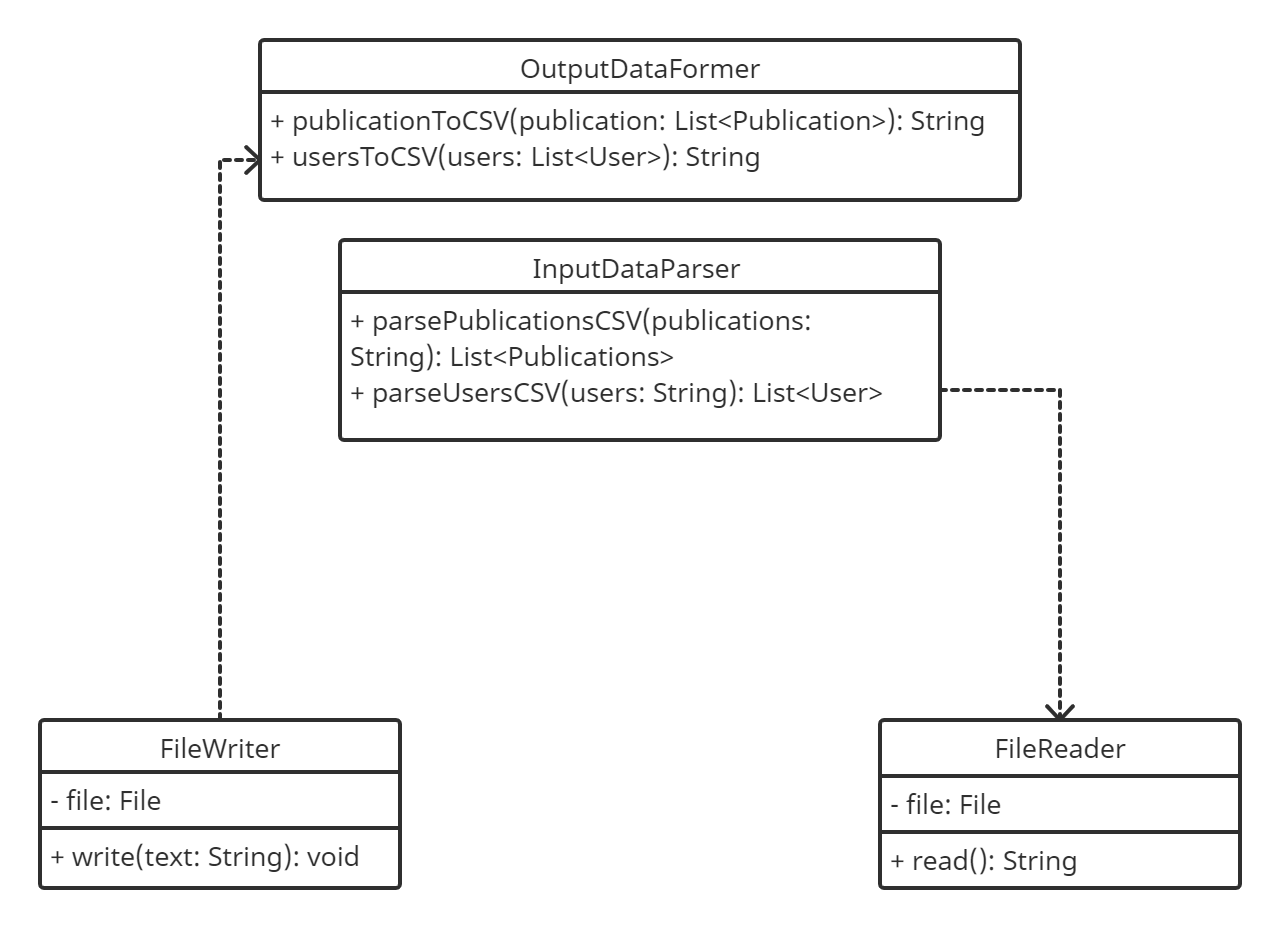


Рисунок 2.3 – UML-діаграма класів формування та парсингу даних з файлів

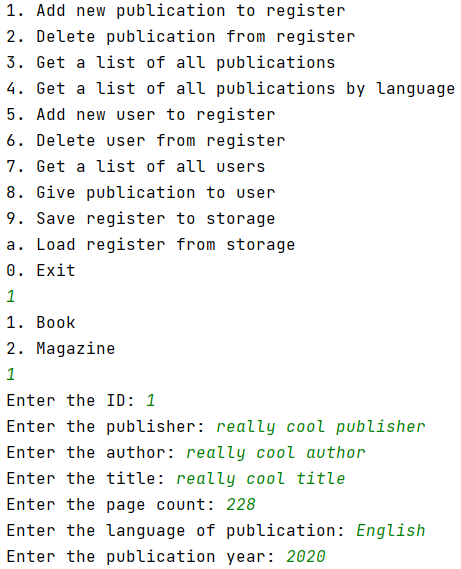
Повну UML-діаграму класів можна подивитися за посиланням 2 в переліку посилань [2].

Повний вихідний код класів можна подивитися в лістингах 1-7

# РОЗДІЛ 3 РЕЗУЛЬТАТИ РОБОТИ

## 3.1 Тестування

Одним із найважливіших етапів в розробці програмного забезпечення є тестування продукту. В ході тестування помилок виявлено не було. Результати тестування приведені на рисунках 3.1 – 3.9.



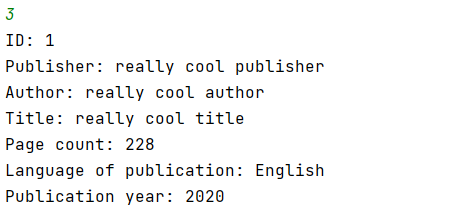
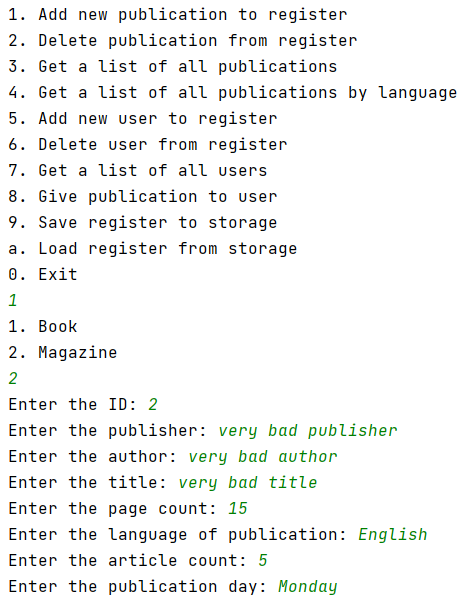


Рисунок 3.1 – Додавання книги в реєстр



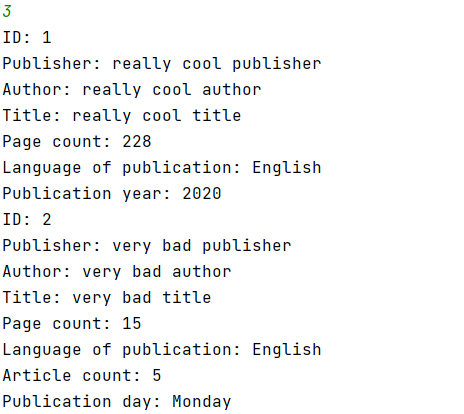
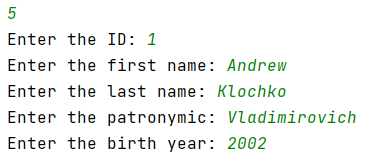


Рисунок 3.2 – Додавання журналу в реєстр



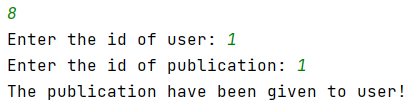


Рисунок 3.3 – Створення нового користувача та видача йому книги

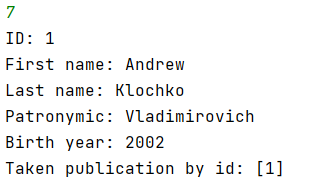


Рисунок 3.4 – Відображення реєстру користувачів бібліотеки

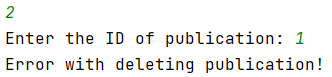


Рисунок 3.5 – Спроба видалення публікації, яка знаходиться не в бібліотеці



Рисунок 3.6 – Збереження реєстру користувачів до файлу users.csv



Рисунок 3.7 – Збереження реєстру публікацій до файлу publications.csv

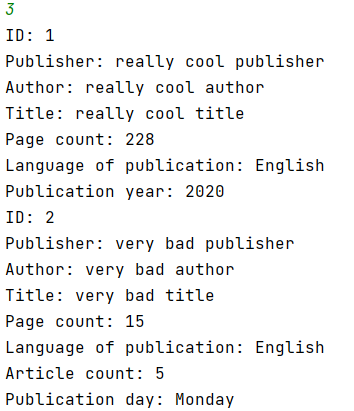


Рисунок 3.8 – Відображення всіх публікацій після завантаження з файлу

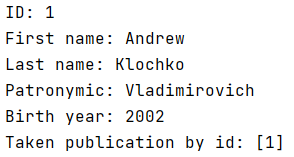


Рисунок 3.9 – Відображення всіх користувачів після завантаження з файлу

## 3.2 Інструкція для користувача

В CLI даної програми присутні такі пункт меню:

1. Add new publication to register
2. Delete publication from register
3. Get a list of all publications
4. Get a list of all publications by language
5. Add new user to register
6. Delete user from register
7. Get a list of all users
8. Give publication to user
9. Save register to storage
10. Load register from storage
11. Exit

Add new publication to register – Додавання нової публікації до реєстру. Не дозволяється додавати публікації з однаковими ID.

Delete publication from register – Видалення існуючої публікацію з регістру. Видалення відбувається по ID публікації. При невірному ID ніяка публікація не видалиться.

Get a list of all publications – Відображення всіх публікацій що містяться в регістрі.

Get a list of all publications by language – Відображення всіх публікацій що містяться в регістрі відфільтровані по мові друку.

Add new user to register – Додавання нового користувача до реєстру. Не дозволяється додавати користувачів з однаковими ID.

Delete user from register – Видалення існуючого користувача з регістру. Видалення відбувається по ID користувача. При невірному ID ніякий користувач не видалиться.

Get a list of all users – Відображення всіх користувачів що містяться в регістрі.

Give publication to user – Видача публікації користувачу. Видача відбувається по ID публікації та ID користувача. При невірних ID видача не відбудеться.

Save register to storage – Збереження реєстру до файлу в форматі .csv.

Load register from storage – Загрузка реєстру з файлу формату .csv.

Exit – Вихід з програми. При виході з програми, не відбувається автозбереження даних до реєстру.

# ВИСНОВОК

В ході виконання курсової роботи було розроблений прототип системи, а саме бібліотеки. Програма містить форми, які дають змогу користувачеві додавати, видаляти, та отримувати необхідну інформацію. Для збереження даних було створено файл, який має розширення .csv, з якого ми отримували та записували інформацію в відповідні файли.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. // creately. URL: https://creately.com/. Дата звернення: 16.12.2020
2. // jetbrains. URL: https://www.jetbrains.com/. Дата звернення: 16.12.2020
3. UML-діаграма класів бібліотеки. URL: https://app.creately.com/diagram/bWOikmTkZyj/edit. Дата звернення: 16.12.2020

# ДОДАТОК А. ВИХІДНИЙ КОД

## Лістинг А.1 – клас Library

package Library.Objects;  
  
import Library.Objects.Persons.User;  
import Library.Objects.Publications.Publication;  
import Library.Objects.Register.PublicationRegister;  
import Library.Objects.Register.UserRegister;  
  
import java.util.List;  
  
public class Library {  
  
 private static final String *NEXT\_LINE* = System.*lineSeparator*();  
  
 private final Address address;  
 private final String libraryName;  
 private final UserRegister userRegister;  
 private final PublicationRegister publicationRegister;  
  
 public Library(String libraryName, Address address){  
 this.address = new Address(address);  
 this.libraryName = libraryName;  
 publicationRegister = new PublicationRegister();  
 userRegister = new UserRegister();  
 }  
  
 public String getLibraryName(){  
 return libraryName;  
 }  
  
 public Address getAddress(){  
 return address;  
 }  
  
 public String getDescription(){  
 return address.getDescription() +  
 "Library name: " +getLibraryName() + *NEXT\_LINE*;  
 }  
  
 */\*--------------------------------------------------------------\*/* public boolean givePublicationToUser(String userId, String publicationId){  
 int indexOfUser = userRegister.indexOf(userId);  
 int indexOfPublication = publicationRegister.indexOf(publicationId);  
  
 if(indexOfUser == -1 || indexOfPublication == -1){  
 return false;  
 }  
  
 if(!publicationRegister.getRegister().get(indexOfPublication).isInLibrary()){  
 return false;  
 }  
  
 userRegister.getRegister().get(indexOfUser).takePublication(publicationId);  
 publicationRegister.getRegister().get(indexOfPublication).setIsInLibraryState(false);  
  
 return true;  
 }  
  
 */\*--------------------------------------------------------------\*/* public void addPublication(Publication publication){  
 publicationRegister.add(publication);  
 }  
  
 public boolean removePublication(String id){  
 return publicationRegister.remove(id);  
 }  
  
 public boolean checkIfPublicationExist(Publication publication){  
 return publicationRegister.checkIfExist(publication);  
 }  
  
 public List<Publication> getPublicationList(){  
 return publicationRegister.getRegister();  
 }  
  
 public void setPublicationList(List<Publication> list){  
 publicationRegister.setList(list);  
 }  
  
 */\*--------------------------------------------------------------\*/* public void addUser(User user){  
 userRegister.add(user);  
 }  
  
 public boolean removeUser(String id){  
 return userRegister.remove(id);  
 }  
  
 public boolean checkIfUserExist(User user){  
 return userRegister.checkIfExist(user);  
 }  
  
 public List<User> getUserList(){  
 return userRegister.getRegister();  
 }  
  
 public void setUserList(List<User> list){  
 userRegister.setRegister(list);  
 }  
  
 */\*--------------------------------------------------------------\*/*}

## Лістинг А.2 – клас PublicationRegister

package Library.Objects.Register;  
  
import Library.Objects.Publications.Publication;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class PublicationRegister {  
  
 private List<Publication> list;  
  
 public PublicationRegister() {  
 this.list = new ArrayList<>();  
 }  
  
 public void add(Publication publication){  
 list.add(publication);  
 }  
  
 public boolean remove(String id){  
 return list.removeIf(p -> (id.equals(p.getId()) && p.isInLibrary()));  
 }  
  
 public boolean checkIfExist(Publication publication){  
 for (Publication p : list) {  
 if(publication.getId().equals(p.getId())){  
 return true;  
 }  
 }  
 return false;  
 }  
  
 public List<Publication> getRegister(){  
 return new ArrayList<>(list);  
 }  
  
 public int indexOf(String id) {  
 for (int i = 0; i < list.size(); i++) {  
 if (list.get(i).getId().equals(id)) {  
 return i;  
 }  
 }  
 return -1;  
 }  
  
 public void setList(List<Publication> list){  
 this.list = new ArrayList<>(list);  
 }  
}

## Лістинг А.3 – абстрактний клас Publication

package Library.Objects.Publications;  
  
public abstract class Publication {  
  
 private final PublicationType type;  
 private final String id;  
 private final String publisher;  
 private final String author;  
 private final String title;  
 private final int pageCount;  
 private final String language;  
 private boolean isInLibrary = true;  
  
 public Publication(PublicationType type, String id, String publisher, String author, String title, int pageCount, String language) {  
 this.type = type;  
 this.id = id;  
 this.publisher = publisher;  
 this.author = author;  
 this.title = title;  
 this.pageCount = pageCount;  
 this.language = language;  
 }  
  
 public Publication(PublicationType type, String id, String publisher, String author, String title, int pageCount, String language, boolean isInLibrary){  
 this.type = type;  
 this.id = id;  
 this.publisher = publisher;  
 this.author = author;  
 this.title = title;  
 this.pageCount = pageCount;  
 this.language = language;  
 this.isInLibrary = isInLibrary;  
 }  
  
 public PublicationType getType(){  
 return type;  
 }  
  
 public String getId(){  
 return id;  
 }  
  
 public String getPublisher(){  
 return publisher;  
 }  
  
 public String getAuthor(){  
 return author;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public int getPageCount() {  
 return pageCount;  
 }  
  
 public String getLanguage() {  
 return language;  
 }  
  
 public boolean isInLibrary() {  
 return isInLibrary;  
 }  
  
 public void setIsInLibraryState(boolean state){  
 isInLibrary = state;  
 }  
  
 public abstract String getDescription();  
}

## Лістинг А.4 – клас Book

package Library.Objects.Publications;  
  
public class Book extends Publication {  
  
 private static final String *NEXT\_LINE* = System.*lineSeparator*();  
  
 private final int publicationYear;  
  
 public Book(String id, String publisher, String author, String title, int pageCount, String language, int publicationYear) {  
 super(PublicationType.BOOK, id, publisher, author, title, pageCount, language);  
 this.publicationYear = publicationYear;  
 }  
  
 public Book(String id, String publisher, String author, String title, int pageCount, String language, boolean isInLibrary, int publicationYear) {  
 super(PublicationType.BOOK, id, publisher, author, title, pageCount, language, isInLibrary);  
 this.publicationYear = publicationYear;  
 }  
  
 public int getPublicationYear() {  
 return publicationYear;  
 }  
  
 public String getDescription(){  
 return "ID: " + getId() + NEXT\_LINE +  
 "Publisher: " + getPublisher() + NEXT\_LINE +  
 "Author: " + getAuthor() + NEXT\_LINE +  
 "Title: " + getTitle() + NEXT\_LINE +  
 "Page count: " + getPageCount() + NEXT\_LINE +  
 "Language of publication: " + getLanguage() + NEXT\_LINE +  
 "Publication year: " + getPublicationYear();  
 }  
}

## Лістинг А.5 – клас Magazine

package Library.Objects.Publications;  
  
public class Magazine extends Publication {  
  
 private static final String *NEXT\_LINE* = System.*lineSeparator*();  
  
 private final int articleCount;  
 private final String publicationDay;  
  
 public Magazine(String id, String publisher, String author, String name, int pageCount, String language, int articleCount, String publicationDay) {  
 super(PublicationType.*MAGAZINE*, id, publisher, author, name, pageCount, language);  
 this.articleCount = articleCount;  
 this.publicationDay = publicationDay;  
 }  
 public Magazine(String id, String publisher, String author, String name, int pageCount, String language, boolean isInLibrary, int articleCount, String publicationDay) {  
 super(PublicationType.*MAGAZINE*, id, publisher, author, name, pageCount, language, isInLibrary);  
 this.articleCount = articleCount;  
 this.publicationDay = publicationDay;  
 }  
  
 public int getArticleCount() {  
 return articleCount;  
 }  
  
 public String getPublicationDay() {  
 return publicationDay;  
 }  
  
 public String getDescription(){  
 return "ID: " + getId() + *NEXT\_LINE* +  
 "Publisher: " + getPublisher() + *NEXT\_LINE* +  
 "Author: " + getAuthor() + *NEXT\_LINE* +  
 "Title: " + getTitle() + *NEXT\_LINE* +  
 "Page count: " + getPageCount() + *NEXT\_LINE* +  
 "Language of publication: " + getLanguage() + *NEXT\_LINE* +  
 "Article count: " +getArticleCount() + *NEXT\_LINE* +  
 "Publication day: " +getPublicationDay();  
 }  
}

## Лістинг А.6 – клас UserRegister

package Library.Objects.Register;  
  
import Library.Objects.Persons.User;  
import Library.Objects.Publications.Publication;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class UserRegister {  
  
 private List<User> list;  
  
 public UserRegister(){  
 list = new ArrayList<>();  
 }  
  
 public void add(User user){  
 list.add(user);  
 }  
  
 public boolean remove(String id){  
 return list.removeIf(u -> (id.equals(u.getId()) && u.getTakenPublicationsId().size() == 0));  
 }  
  
 public boolean checkIfExist(User user){  
 for (User u : list) {  
 if(user.getId().equals(u.getId())){  
 return true;  
 }  
 }  
 return false;  
 }  
  
 public List<User> getRegister(){  
 return new ArrayList<>(list);  
 }  
  
 public int indexOf(String id){  
 for (int i = 0; i < list.size(); i++) {  
 if(list.get(i).getId().equals(id)){  
 return i;  
 }  
 }  
 return -1;  
 }  
  
 public void setRegister(List<User> list){  
 this.list = new ArrayList<>(list);  
 }  
  
}

## Лістинг А.7 – клас User

package Library.Objects.Persons;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class User{  
  
 private static final String NEXT\_LINE = System.lineSeparator();  
  
 private final String id;  
 private final String firstName;  
 private final String lastName;  
 private final String patronymic;  
 private final String birthYear;  
 private final List<String> takenPublicationsId;  
  
  
 public User(String id, String firstName, String lastName, String patronymic, String birthYear) {  
 this.id = id;  
 this.firstName = firstName;  
 this.lastName = lastName;  
 this.patronymic = patronymic;  
 this.birthYear = birthYear;  
 takenPublicationsId = new ArrayList<>();  
 }  
  
 public User(String id, String firstName, String lastName, String patronymic, String birthYear, List<String> takenPublicationsId) {  
 this.id = id;  
 this.firstName = firstName;  
 this.lastName = lastName;  
 this.patronymic = patronymic;  
 this.birthYear = birthYear;  
 this.takenPublicationsId = takenPublicationsId;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public String getFirstName() {  
 return firstName;  
 }  
  
 public String getLastName() {  
 return lastName;  
 }  
  
 public String getPatronymic() {  
 return patronymic;  
 }  
  
 public String getBirthYear(){  
 return birthYear;  
 }  
  
 public List<String> getTakenPublicationsId(){  
 return new ArrayList<>(takenPublicationsId);  
 }  
  
 public void takePublication(String id){  
 takenPublicationsId.add(id);  
 }  
  
 public String getDescription(){  
 return "ID: " + getId() + NEXT\_LINE +  
 "First name: " + getFirstName() + NEXT\_LINE +  
 "Last name: " + getLastName() + NEXT\_LINE +  
 "Patronymic: " + getPatronymic() + NEXT\_LINE +  
 "Birth year: " + getBirthYear() + NEXT\_LINE +  
 "Taken publication by id: " + takenPublicationsId.toString();  
 }  
}