

**Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»**

**ЗВІТ
З ЛАБОРАТОРНОЇ РОБОТИ**

Дисципліна “Поглиблене програмування на Java”

Виконано: Ушенко О. В.,
122-21-4
Перевірено: Мінесв О.С

**Дніпро
2025**

Тема: «Створення проекту на Java. Отримання базових навичок».

Хід роботи

1. Hello world

Встановити IntelliJ Idea та Java jdk останньої версії. Створити maven проект та розробити в цьому проекті типову програму Hello world. Програма повинна видавати на екран напис Hello world та закінчувати свою роботу. Під'єднати до intelliJ Idea систему CVS. А саме GIT. Створити аккаунт в хмарному середовищі github, під'єднати свій проект в intelliJ Idea до свого аккаунту github та завантажити нульову лабораторну роботу на github аккаунт. Кожну нову лабораторну роботу робити в окремій гілці(з іменем лабораторної наприклад «LR_3») а потім після того як її написали мержити гілку до мастера.

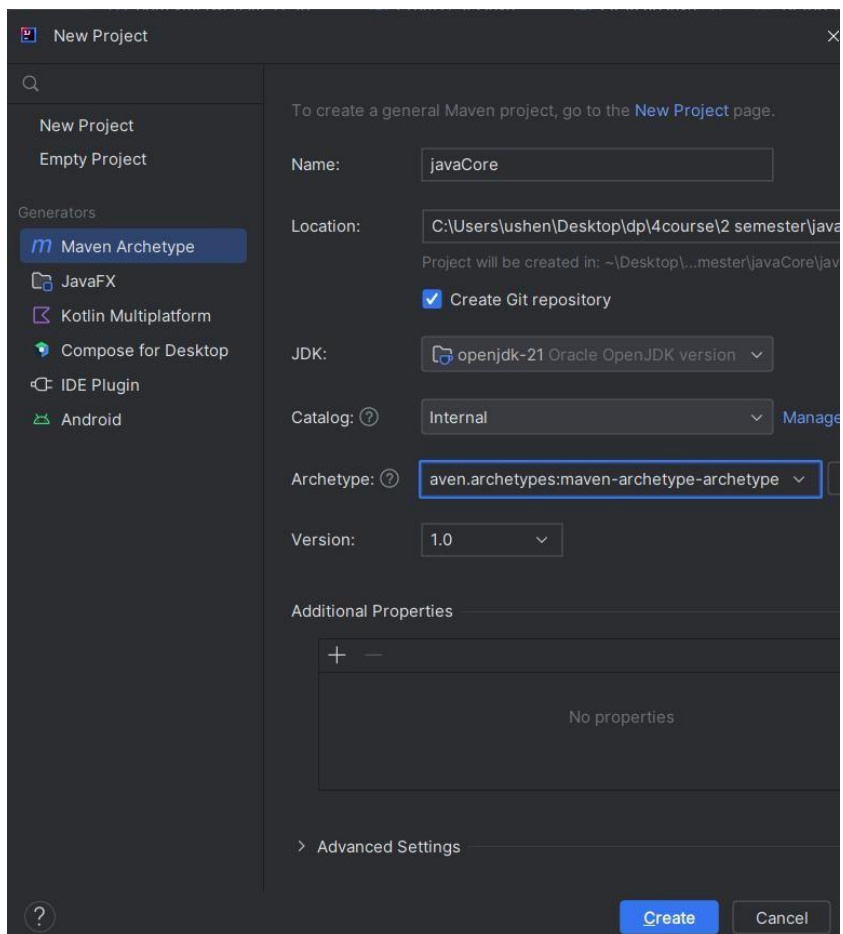





Рис 1. Створення проекту


```
Run hello x
C:\Users\ushen\.jdk\openjdk-21\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.2\lib\idea_rt
hello
Process finished with exit code 0
```


Рис 2. Результат програми Hello World


 Public


 Pin

 Unwatch 1



 Fork 0

 Star 0

**Set up GitHub Copilot**
Use GitHub's AI pair programmer to autocomplete suggestions as you code.
[Get started with GitHub Copilot](#)

**Add collaborators to this repository**
Search for people using their GitHub username or email address.
[Invite collaborators](#)

Quick setup — if you've done this kind of thing before

 Set up in Desktop or [HTTPS](#) [SSH](#) 

Terminal Local x + v

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git remote add origin <https://github.com/helena-78/javaCore.git>
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git branch -M main
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push -u origin main
error: src refspec main does not match any
error: failed to push some refs to '<https://github.com/helena-78/javaCore.git>'
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git status
On branch main

No commits yet

Changes to be committed:
 create mode 100644 src/main/resources/archetype-resources/src/test/java/AppTest.java
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push -u origin main
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 12 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (24/24), 2.84 KiB | 728.00 KiB/s, done.
Total 24 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To <https://github.com/helena-78/javaCore.git>
 * [new branch] main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> |

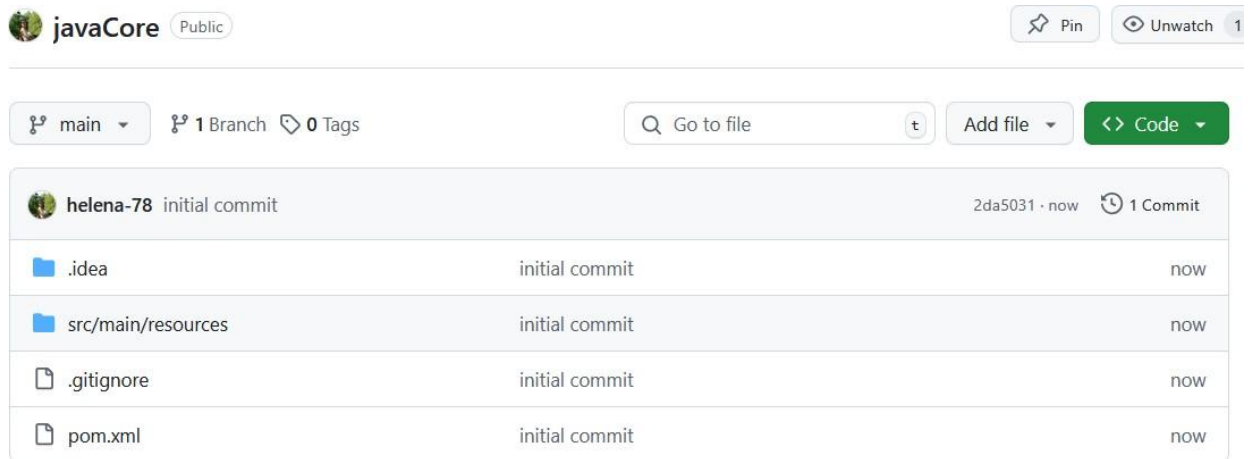


Рис 3 – 5. Створення репозиторія

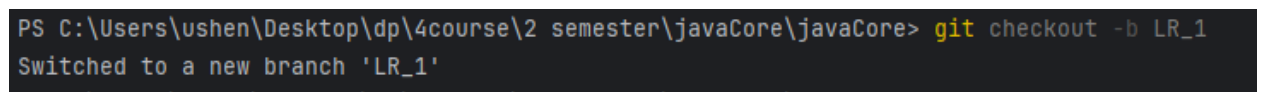
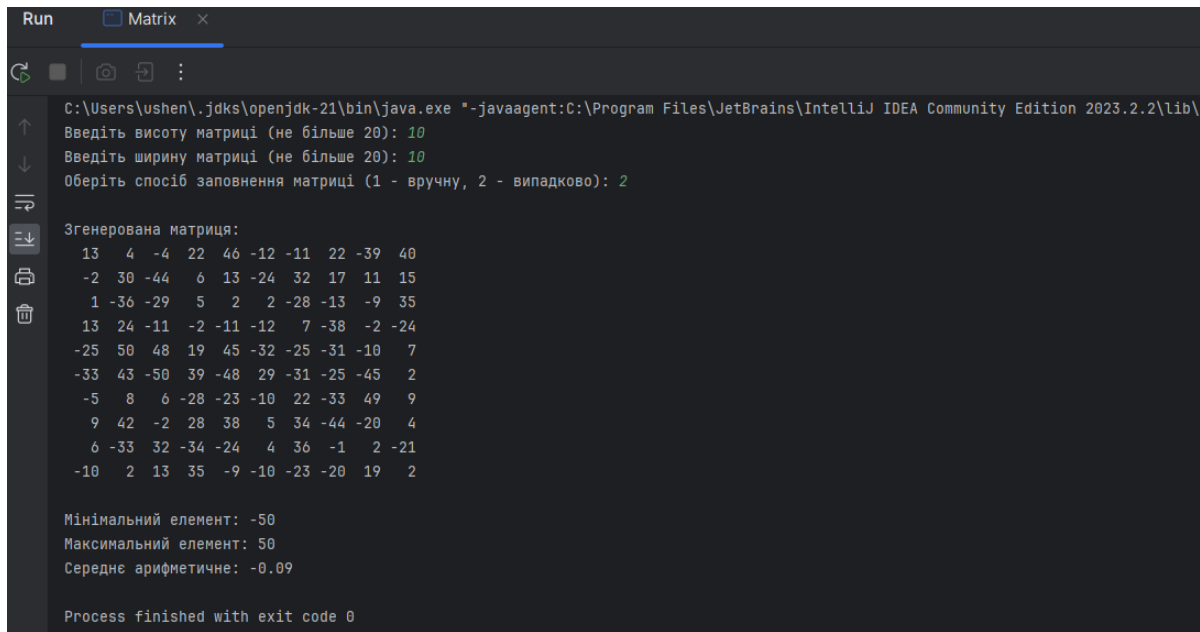


Рис 6. Розміщення лр0 у окрему гілку LR_0

2. Основи.

Розробити програму, що дозволить вам створити, як з клавіатури так і рандомно матрицю цілих чисел типу `int` заданої ширини та висоти(ввести з клавіатури), але не більше 20 на 20. Створити можливість пошуку в цій матриці мінімального і максимального елементу та розрахунок середнього арифметичного. Програма може бути написана в одному класі, обов'язково розбиття на методи. Обов'язкове використання клавіатури, під час вибору ручного чи рандомного створення матриці. Створення системи зчитування з клавіатури зробити будь-яким способом, наприклад завдяки класу `Scanner`. `Scanner` являє собою найпростішу систему сканування клавіатури. Діапазон рандомних чисел для створення елементів матриці повинен зверігатись в спеціальних константах.



```
Run Matrix x
C:\Users\ushen\.jdk\openjdk-21\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.2\lib\
Введіть висоту матриці (не більше 20): 10
Введіть ширину матриці (не більше 20): 10
Оберіть спосіб заповнення матриці (1 - вручну, 2 - випадково): 2

Згенерована матриця:
13  4 -4 22 46 -12 -11 22 -39 40
-2 30 -44 6 13 -24 32 17 11 15
1 -36 -29 5 2 2 -28 -13 -9 35
13 24 -11 -2 -11 -12 7 -38 -2 -24
-25 50 48 19 45 -32 -25 -31 -10 7
-33 43 -50 39 -48 29 -31 -25 -45 2
-5 8 6 -28 -23 -10 22 -33 49 9
9 42 -2 28 38 5 34 -44 -20 4
6 -33 32 -34 -24 4 36 -1 2 -21
-10 2 13 35 -9 -10 -23 -20 19 2

Мінімальний елемент: -50
Максимальний елемент: 50
Середнє арифметичне: -0.09

Process finished with exit code 0
```

Рис 7. Результат програми



Рис 8. Розміщення lr2 у окрему гілку LR_2

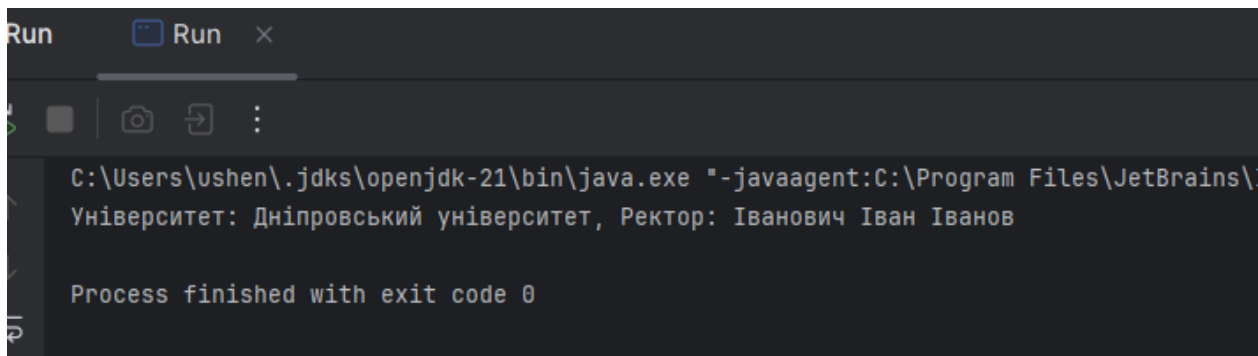
3. ООП

Створити програму що буде створювати та обробляти комплексний об'єкт під назвою університет(university). Програма повинна складатися з трьох частин: модель вид та контролер згідно з парадигмою mvc (Model View Controller). Кожній з цих груп повинна відповідати package з відповідною назвою. В моделі повинні знаходитись усі класи що відповідають за структурні підрозділи університету. Серед них: університет, факультет, кафедра, група, студент, людина (Human). Усі вони повинні містити назву типу string та голову типу Human. Студент також повинен бути породжений від Human. Human повинен мати поля ім'я, прізвище, побатькові та стать. Усі поля повинні бути строковими окрім поля стать. Стать повинна використовувати спеціальний enum типу Sex(стать).

В цій лабораторній роботі група View Нам не потрібна.

Що стосується групи контроллер (controller) то вона повинна містити менеджери що дозволяють нам створити відповідні підрозділи наприклад StudentCreator, FacultyCreator, GroupCreator та інші, кожен з яких повинен використовувати можливості нижчого за рівнем створювача. Програма повинна також містити клас Run, в якому буде знаходитись точка входу та методи, що повинні дати можливість створити університет. Процес створення університету повинен бути зроблений в методі createTypicalUniversity.

В програмі активно рекомендується використовувати абстрактні класи та інтерфейси



```
Run
C:\Users\ushen\.jdk\openjdk-21\bin\java.exe -javaagent:C:\Program Files\JetBrains\
Університет: Дніпровський університет, Ректор: Іванович Іван Іванов
Process finished with exit code 0
```

Рис 9. Результат програми

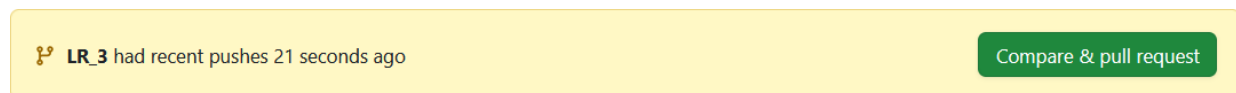


Рис 10. Розміщення лр3 у окрему гілку LR_3

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git merge LR_0
Updating 2da5031..4e4883d
Fast-forward
 src/main/java/Hello.java | 5 +++++
 1 file changed, 5 insertions(+)
 create mode 100644 src/main/java/Hello.java

```

Рис 11. Успішний мердж LR_0 з main

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git merge LR_2
Merge made by the 'ort' strategy.
 src/main/java/Matrix.java | 155 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 155 insertions(+)
 create mode 100644 src/main/java/Matrix.java

```

Рис 12. Успішний мердж LR_2 з main

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git merge LR_3
error: The following untracked working tree files would be overwritten by merge:
    .idea/uiDesigner.xml
Please move or remove them before you merge.
Aborting
Merge with strategy ort failed.

```

Рис 13. Провальний мердж LR_3 з main

В робочому каталозі є непроіндексований файл `.idea/uiDesigner.xml`, який буде перезаписаний при злитті. Щоб продовжити, потрібно вирішити цю проблему.

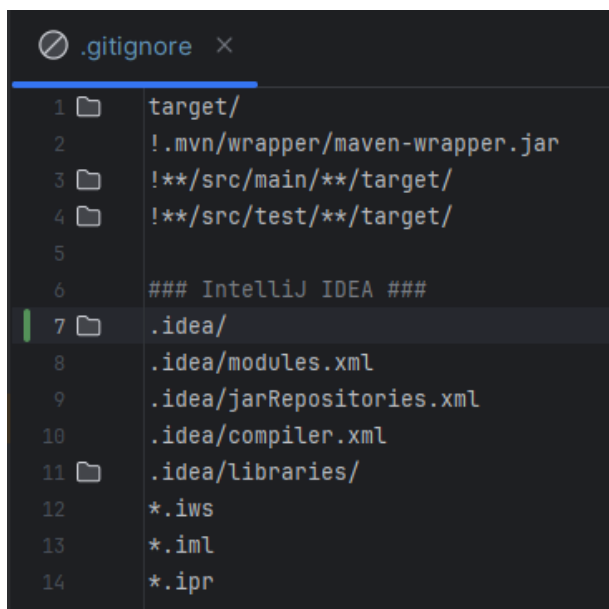
Додаємо файл до комміту.

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git add .idea/uiDesigner.xml
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git commit -m "add uiDesigner.xml before merge"
[main 74d8dbf] add uiDesigner.xml before merge
1 file changed, 124 insertions(+)
create mode 100644 .idea/uiDesigner.xml
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git merge LR_3
Merge made by the 'ort' strategy.
 pom.xml | 4 +++
src/main/java/controller/DepartmentCreator.java | 10 ++++++
src/main/java/controller/FacultyCreator.java | 10 ++++++
src/main/java/controller/GroupCreator.java | 10 ++++++
src/main/java/controller/Run.java | 32 ++++++
src/main/java/controller/StudentCreator.java | 10 ++++++
src/main/java/controller/UniversityCreator.java | 10 ++++++
src/main/java/model/Department.java | 25 ++++++
src/main/java/model/Faculty.java | 25 ++++++
src/main/java/model/Group.java | 25 ++++++
src/main/java/model/Human.java | 19 ++++++
src/main/java/model/Sex.java | 5 +++
src/main/java/model/Student.java | 12 ++++++
src/main/java/model/University.java | 25 ++++++
14 files changed, 222 insertions(+)

```

Рис 14. Успішний мердж LR_3 з main



```

.
target/
!.mvn/wrapper/maven-wrapper.jar
!**/src/main/**/target/
!**/src/test/**/target/
### IntelliJ IDEA ###
.idea/
.idea/modules.xml
.idea/jarRepositories.xml
.idea/compiler.xml
.idea/libraries/
*.iws
*.iml
*.ipr

```

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git add .gitignore
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git commit -m "add .idea/ to .gitignore"
[main 689ab2e] add .idea/ to .gitignore
1 file changed, 1 insertion(+)

```

Рис 15 – 16. Щоб уникнути цієї проблеми у майбутньому, додаємо .idea/ у .gitignore

4. JUnit. Json

Додати до лабораторної роботи 3 можливість запису університету у формат json, запис цього формату у файл, зчитування цього формату файлу, та створення об'єкту з текстового формату json. В проекті повинен бути зроблений JUnit тест, який буде виглядати наступним чином: створити об'єкт університет(oldUniversity), в якому в кожному підрозділі маються два підрозділи нижчого рівня. Наприклад на факультеті дві кафедри, на кожній кафедрі дві групи, на кожній групі два студенти. Цей об'єкт повинен бути записаний в файл у форматі json. Потім з цього файлу зчитаний та відновлений як newUniversity. В тесті повинні бути порівняні newUniversity та oldUniversity за допомогою методу equals. Якщо все зроблено правильно то університети повинні бути еквівалентні, а метод equals повинен повернути True. Для запису та зчитування університету у форматі json повинен бути зроблений клас JsonManager. Для безпосереднього перетворення університету у формат json та його відновлення цього формату, можливо використання сторонніх бібліотек наприклад Gson, Jackson чи будь-яких інших.

Для початку розробки лабораторної роботи номер 4 повністю скопіювати програмний код лабораторної роботи номер 3. Не змішувати ці роботи не в якому разі.

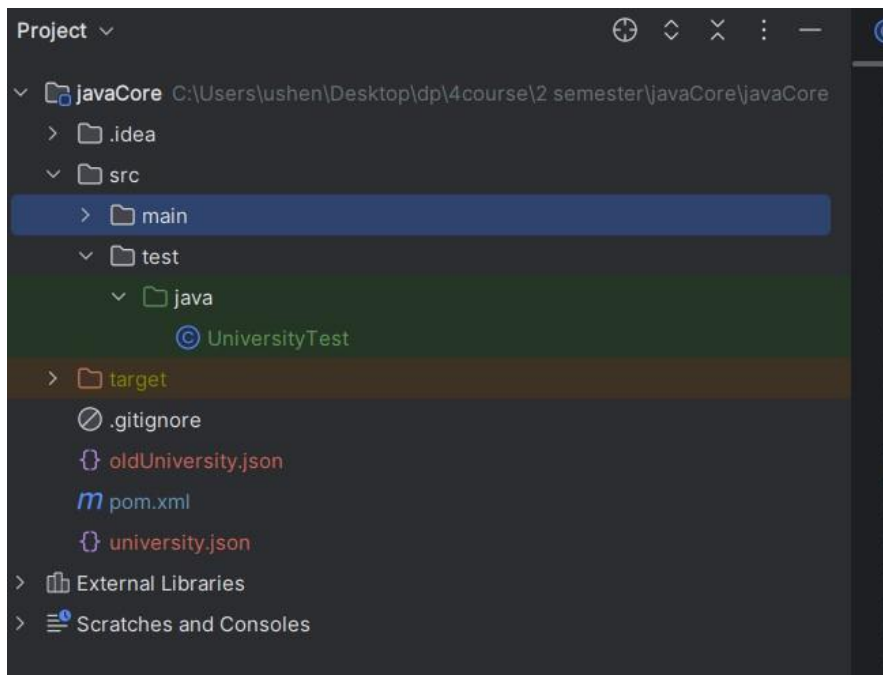


Рис 17. Створення тесту JUnit

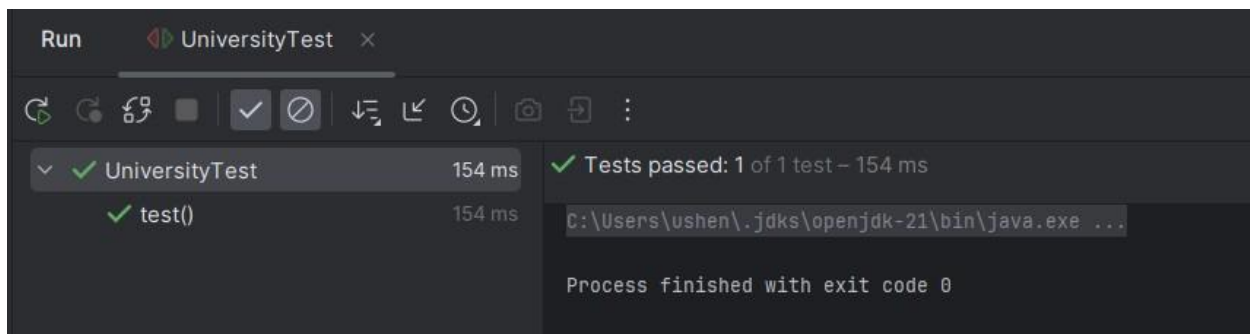


Рис 18. Результат програми

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 7 commits.
  (use "git push" to publish your local commits)
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git pull origin main
From https://github.com/helena-78/javaCore
* branch          main          -> FETCH_HEAD
Already up to date.
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git merge LR_4
Updating 689ab2e..9c29ee0
Fast-forward
 oldUniversity.json      | 1 +
 pom.xml                 | 22 +++++
 .../java/lab4/controller/DepartmentCreator.java | 10 ++
 src/main/java/lab4/controller/FacultyCreator.java | 10 ++
 src/main/java/lab4/controller/GroupCreator.java   | 10 ++
 src/main/java/lab4/controller/Run.java            | 39 ++++++++
 src/main/java/lab4/controller/StudentCreator.java | 10 ++
 .../java/lab4/controller/UniversityCreator.java   | 10 ++
 src/main/java/lab4/json/JsonManager.java          | 32 ++++++++
 src/main/java/lab4/model/Department.java          | 51 ++++++++
 src/main/java/lab4/model/Faculty.java             | 51 ++++++++
 src/main/java/lab4/model/Group.java               | 51 ++++++++
 src/main/java/lab4/model/Human.java               | 40 ++++++++
 src/main/java/lab4/model/Sex.java                 | 5 +
 src/main/java/lab4/model/Student.java             | 33 ++++++++
 src/main/java/lab4/model/University.java          | 52 ++++++++
 src/test/java/UniversityTest.java                 | 101 ++++++++
 university.json      | 1 +
18 files changed, 529 insertions(+)
create mode 100644 oldUniversity.json

```

Рис 19. Розміщення лр4 у окрему гілку LR_4

```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push origin --delete LR_4
To https://github.com/helena-78/javaCore.git
- [deleted]          LR_4
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push origin --delete LR_0
To https://github.com/helena-78/javaCore.git
- [deleted]          LR_0
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push origin --delete LR_1
error: unable to delete 'LR_1': remote ref does not exist
error: failed to push some refs to 'https://github.com/helena-78/javaCore.git'
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push origin --delete LR_2
To https://github.com/helena-78/javaCore.git
- [deleted]          LR_2
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git push origin --delete LR_3
To https://github.com/helena-78/javaCore.git
- [deleted]          LR_3
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore>

```

Рис 20. Мерджемо гілку, видаляємо непотрібні гілки LR_0 - 4

5. Jdbc

Створити базу даних в будь якому сервері баз даних. Створити таблицю з переліком студентів вказати їх прізвище, ім'я, по батькові, день народження номер залікової книжки та ID. Створити програму що буде дозволяти виводити на екран інформацію про студентів які народилися в тому чи іншому місяці року. Програма повинна завдяки системі jdbc під'єднатися до вашої бази даних та робити до неї запити. Вимог до розробки бази даних немає. Програма ж має бути написана за усіма стандартами ООП. Та може бути спроектована за двох принципів:

- при будь-якій ситуації буде забиратися весь перелік студентів, а вже на стороні java буде зроблено пошук необхідного
- SQL запит буде сформований згідно запиту який зробив користувач і вже сервер управління баз даних буде вирішувати, які самі студенти народилися в тому чи іншому місяці.

У висновку обов'язково пояснити чому вибрали той чи інший принцип, які в нього переваги та недоліки. Оцінка не залежить від того який сервер управління баз даних вибрали. Перелік студентів зробити не менше 20 людей. Місяць червень зробити місяцем, коли в жодного зі студентів немає дня народження.

SQL код створення бази даних розмістити проєкті 6 лабораторної роботи в файлі database в папечці resources. Для використання цієї лабораторної роботи

рекомендується активно використовувати знання отримані на дисципліні що стосуються розробки баз даних.

До паперового звіту обов'язково додати принтскрин з програми в якій ви дивитесь інформацію вашого сервера управління баз даних, де показати створену таблицю, її ім'я та загальні відомості бази даних, наприклад назва, ім'я, назва користувача адміністратора, пароль тощо. Для роботи з сервером управління баз даних рекомендуємо використовувати програмне забезпечення компанії JetBrains DataGrip. Або вбудовану панель користування базами даних, що міститься у середовищі IntelliJ IDEA, яка на сьогоднішній день підтримує майже всі сервери управління баз даних.

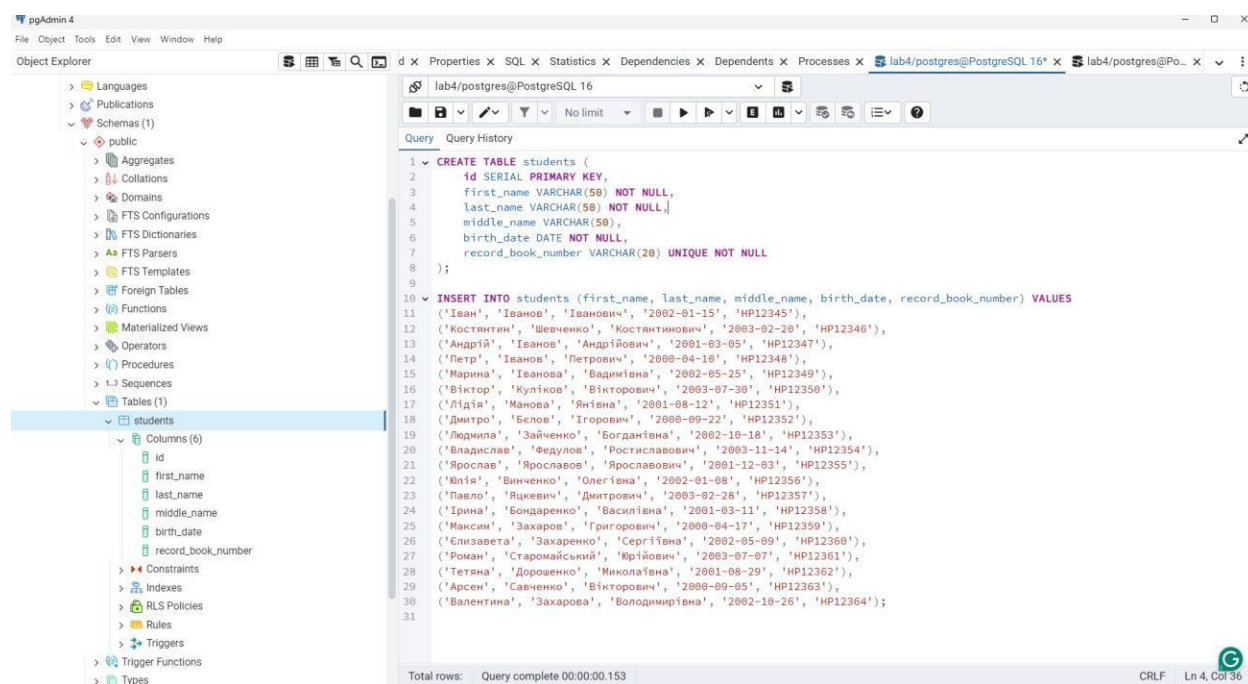


Рис 21. Створення бази даних postgresSQL

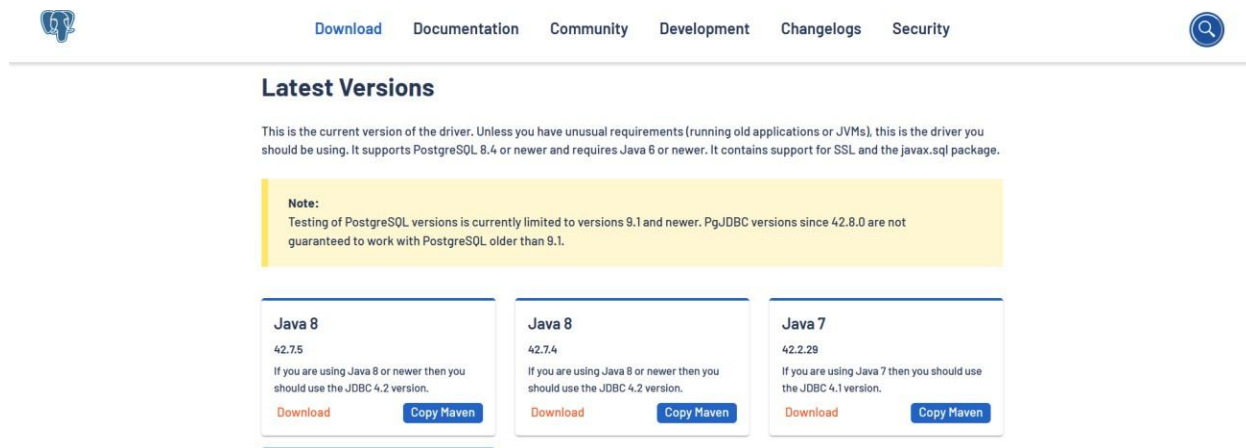


Рис 22. Завантажуємо драйвер postgresSQL JDBC

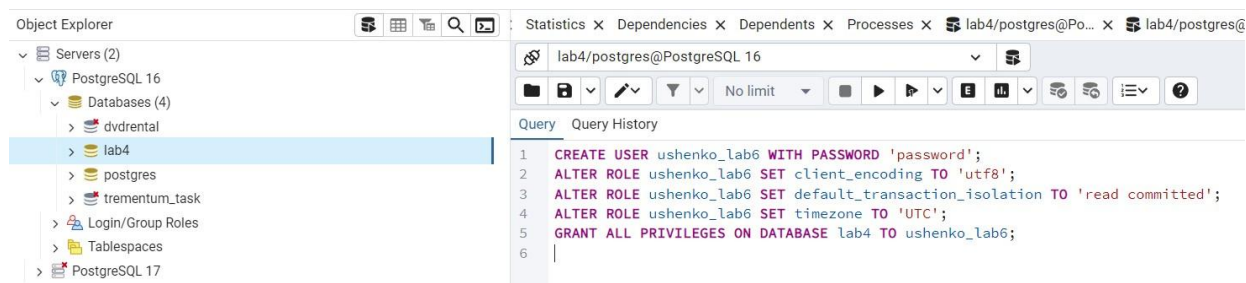


Рис 23. Створюємо користувача щоб підключитися до бази даних

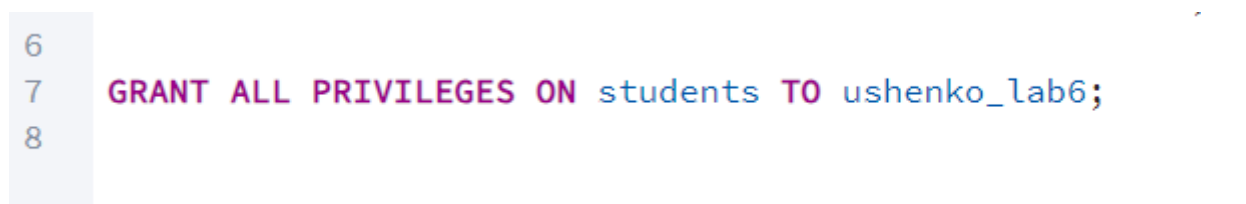


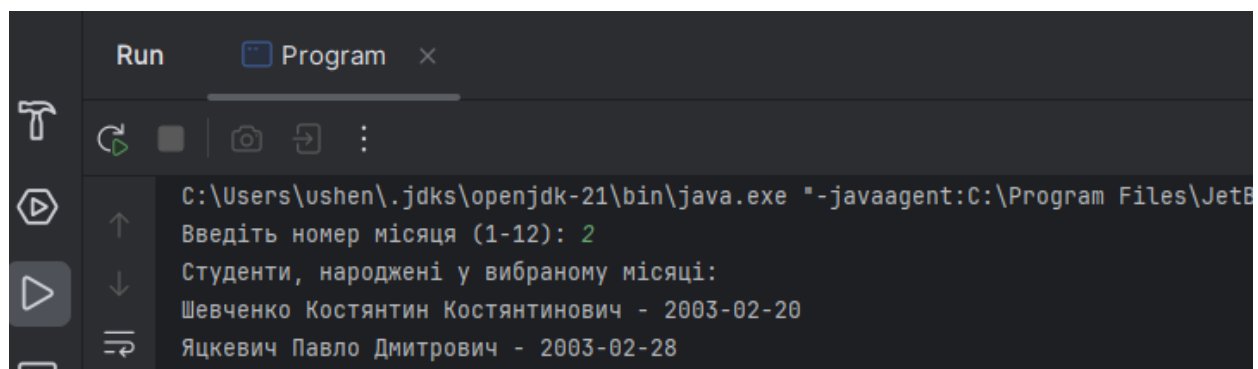
Рис 24. Надаємо всі права користувачу ushenko_lab6

```

18     </plugin>
19   </plugins>
20 </build>
21
22   <dependencies>
23     <dependency>
24       <groupId>org.postgresql</groupId>
25       <artifactId>postgresql</artifactId>
26       <version>42.6.0</version>
27     </dependency>
28   </dependencies>
29
30 </project>
31

```

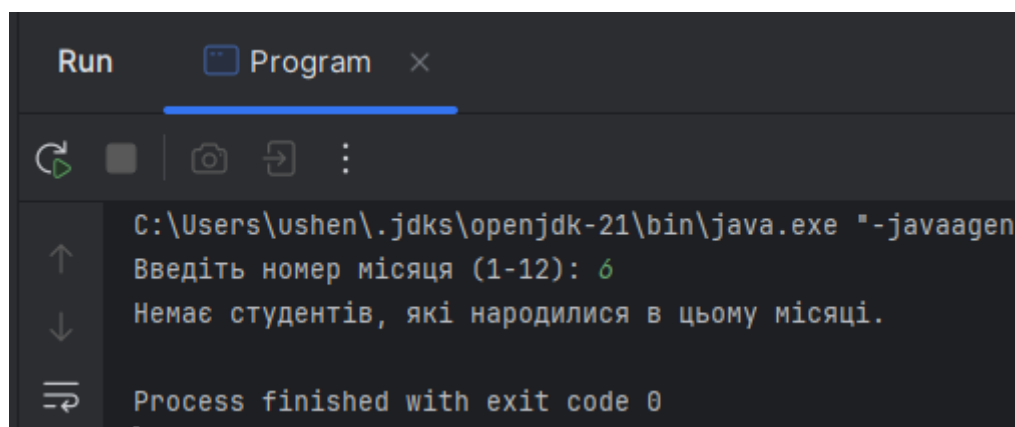
Рис 25. Підключаємо драйвер postgres до intellijidea.



```

Run   Program  x
C:\Users\ushen\.jdk\openjdk-21\bin\java.exe "-javaagent:C:\Program Files\JetB
Введіть номер місяця (1-12): 2
Студенти, народжені у вибраному місяці:
Шевченко Костянтин Костянтинович - 2003-02-20
Яцкевич Павло Дмитрович - 2003-02-28

```



```

Run   Program  x
C:\Users\ushen\.jdk\openjdk-21\bin\java.exe "-javaagen
Введіть номер місяця (1-12): 6
Немає студентів, які народилися в цьому місяці.
Process finished with exit code 0

```

Рис 26 – 27. Результат програми


```

PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\ushen\Desktop\dp\4course\2 semester\javaCore\javaCore> git merge LR_5
Updating 6023dc3..b0456c2
Fast-forward
 lab5/pom.xml | 30 ++++++
 lab5/src/main/Program.java | 19 +++++
 .../main/database/resources/DatabaseManager.java | 36 ++++++
 .../main/resources/META-INF/maven/archetype.xml | 9 +++++
 .../src/main/resources/archetype-resources/pom.xml | 15 +++++
 .../archetype-resources/src/main/java/App.java | 13 +++++
 .../archetype-resources/src/test/java/AppTest.java | 38 ++++++
 7 files changed, 160 insertions(+)
 create mode 100644 lab5/pom.xml
 create mode 100644 lab5/src/main/Program.java
 create mode 100644 lab5/src/main/database/resources/DatabaseManager.java

```

Рис 28. Створюємо нову гілку LR_5, додаємо зміни до github та мерждемо LR_5 з main

Висновок

Стосовно п'ятої практичної роботи, було два варіанти – SQL фільтрація на сервері або отримання всіх студентів на java. Був обран перший варіант, тому що запит обробляється швидше, менше навантаження на пам'ять java. Також, postgresql краще працює з фільтрацією ніж java.

Під час виконання всіх практичних робіт було засвоєно базові знання java, такі як абстрактні класи, ООП, тестування з JUnit 5.