**ЛАБОРАТОРНА РОБОТА № 1**

**Тема:** Знайомство з мовою програмування Java. Написання простих програм на мові програмування Java

**Мета роботи:** Вивчити реалізацію базових алгоритмічних конструкцій у мові програмування Java; знайомство з правилами оформлення програмного коду

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Написання простих програм:**

**Програма 1:**

Ім'я класу: com.education.ztu.Task1

Напишіть клас, який реалізує функціональність відображення рядка «Hello, World!!!» у консолі.



**Програма 2**

Ім'я класу: com.education.ztu.Task2

Напишіть клас, який реалізує функціональність додавання двох цілих чисел. Для зчитування даних використовувати методи класу Scanner.



**Програма 3**

Ім'я класу: com.education.ztu.Task3

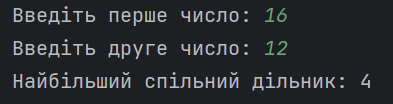
Напишіть клас, який реалізує функціональність відображення параметрів командного рядка в консолі (відображення через пробіл між ними), результат не повинен закінчуватися пробілом. Аргументи передавати таким чином Task3.main(new String[]{"2", "3", "5", "8"}); в класі Main.



**Програма 4**

Ім'я класу: com.education.ztu.Task4

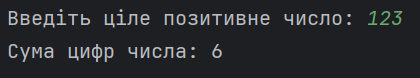
Напишіть клас, який реалізує функціональні можливості визначення найбільшого спільного дільника двох цілих додатних чисел. Для зчитування даних використовувати методи класу Scanner.



**Програма 5**

Ім'я класу: com.education.ztu.Task5

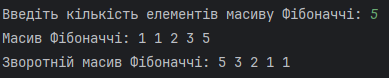
Напишіть клас, який реалізує функціональні можливості визначення суми цифр цілого позитивного числа. Для зчитування даних використовувати методи класу Scanner.



**Програма 6**

Ім'я класу: com.education.ztu.Task6

Напишіть клас, який створює масив із n елементів і заповнює його зростаючою послідовністю чисел Фібоначчі (1,1,2,3,5,8…). Створити новий масив та заповнити його зворотньою послідовністю Фібоначчі. Вивести в консоль обидва масиви. Для зчитування даних використовувати методи класу Scanner.



**Програма 7**

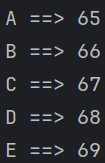
Ім'я класу: com.education.ztu.Task7

Створити масив символів латинського алфавіту та вести їх числові коди в такому форматі:

A ==> 65

B ==> 66

C ==> 67



**Висновок:** Я вивчив реалізацію базових алгоритмічних конструкцій у мові програмування Java; ознайомився з правилами оформлення програмного коду

**ЛАБОРАТОРНА РОБОТА № 2**

**Тема:** Створення структури класу заданої предметної області.

**Мета роботи:** Створити ієрархію класів заданої предметної області, робота з статичними методами.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Завдання 2.** Створити ієрархію класів відповідно до UML діаграми:

- Поля класів повинні бути приховані модифікаторами доступу private, protected;

- Створити конструктор без аргументів та з агрументами;

- Чтворити блок ініціалізації, в якому ініціалізуються значення полів за замовчуванням у разі, якщо викликається конструктор без аргументів;

- Створити геттери та сеттери для полів;

- Створити статичну змінну counter для підрахунку створених екземплярів даного класу та статичний метод showCounter для відображення значення змінної counter.

- Створити enam Location та Gender і використати їх в полях класів.

- Створити інтерфейс Human з методами sayFullName, sayAge, sayLocation, sayGender та whoIAm (default) .

- Створити абстрактний клас Person з абстрактним методом getOccupation та звичайним методом getFullInfo, що імплементує Human;

- Створити класу Student, Teacher, Employee, що наслідують Person та перевизначити необхідні методи та створити свої.

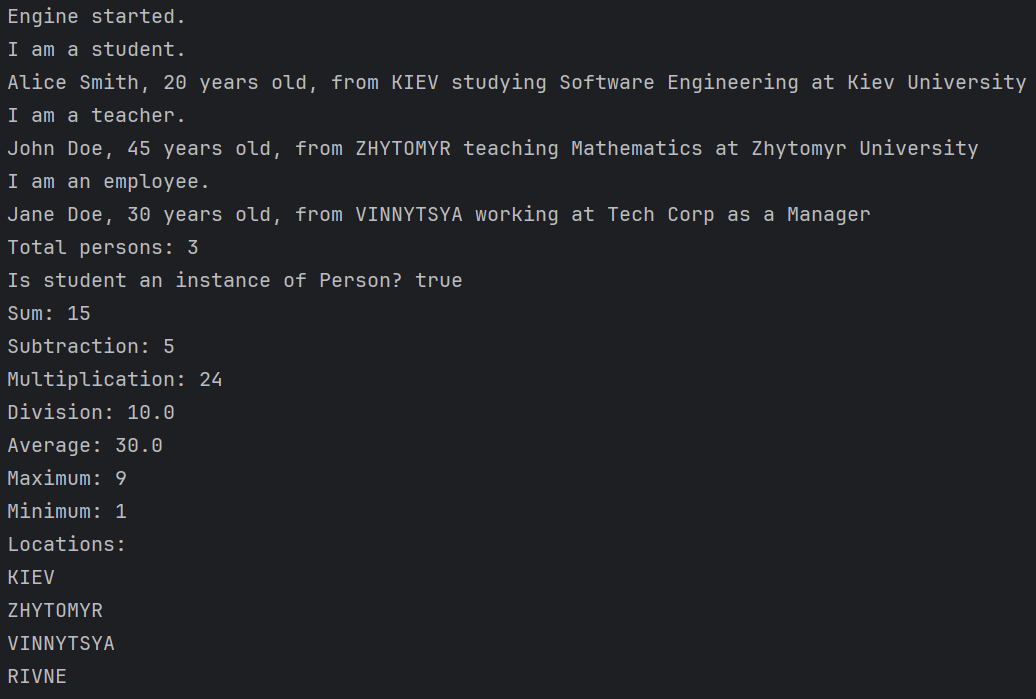
- Для Teacher, Employee додати поле Car , що є об'єктом відповідного класу.

- Створити в Car внутрішній клас Engine з методами startEngine, stopEngine, isEngineWorks та реалізувати їх логіку.

- Додати до описаної функціональності свою (нові поля та методи).

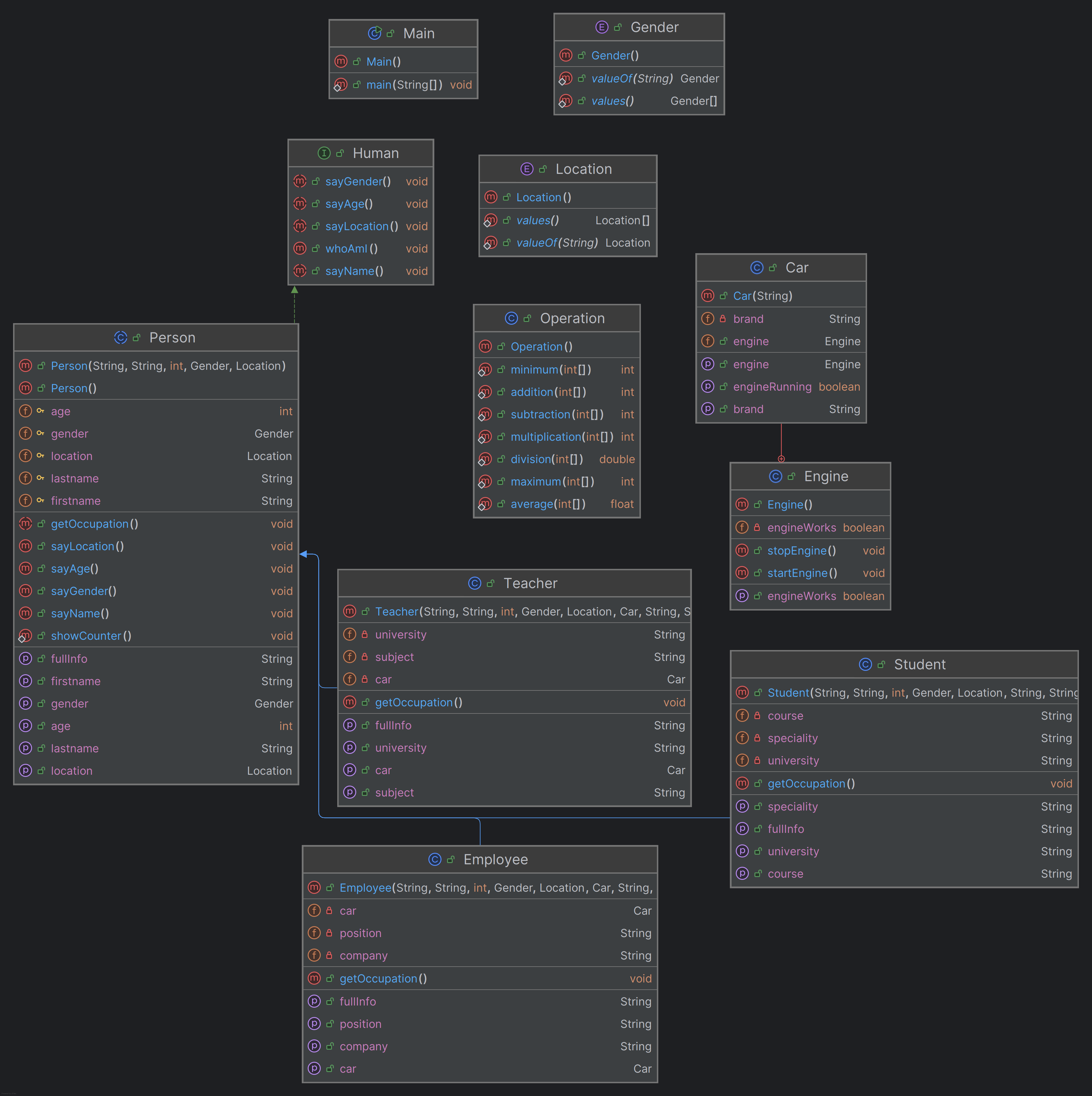
- В методі main класу Main створити об'єкти відповідних класів та продемонструвати роботу їх методів.

- продемонтрувати роботу оператору instanceof.



**Завдання 3.** Створити клас Operation з статичними методами addition, subtraction, multiplication, division, average, maximum, minimum, що приймають необмежену кількість аргументів через varargs. в методі main класу Main2 продемонструвати роботу методів класу Operation

- вивести всі значення enam Location.



**Висновок:** Я створив ієрархію класів заданої предметної області, працював з статичними методами

**ЛАБОРАТОРНА РОБОТА № 3**

**Тема:** Використання узагальнень (generics). Клонування та порівняння об'єктів.

**Мета роботи:** Створити міні проект Game з використанням узагальнень, клонування та порівняння об'єктів.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Завдання 1.** Відкрити заготовлений проект з реалізованою базовою функціональністю.

**Завдання 2.** За допомогою узагальнень (generics) встановити такі обмеження:   
- до команди можна додавати тільки учасників, що відносяться до одної ліги (Schoolar, Student або Employee).

- грати між собою можуть тільки команди з учасниками одної ліги (тобто команда студентів може грати тільки іншою командою студентів).

- продемонструвати створення команд, гравців, додавання гравців до команд, гри між ними.

**Завдання 3.** Клонування:

- для класу Participant імплементувати інтерфейс Cloneable та перевизначити метод clone.

- для класу Participant перевизначити методи hashCode та equals.

- для класу Participant та його підкласів перевизначити метод toString.

- для класу Team Реалізувати глибоке клонування через статичний метод або конструктор копіювання.

- продемонструвати клонування та використання методів hashCode, equals та toString.

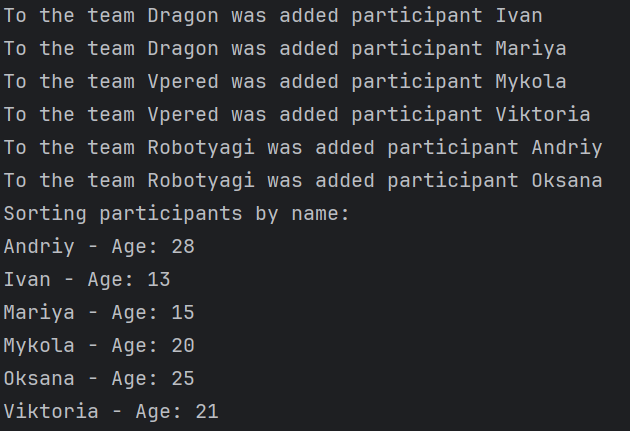
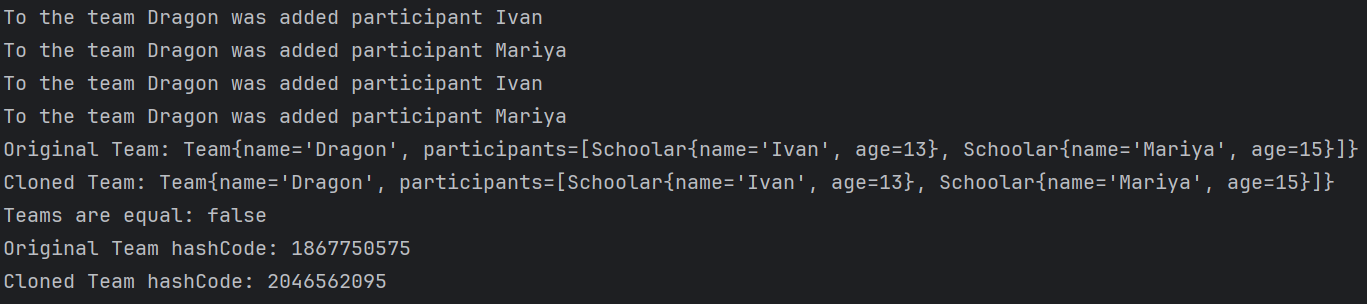
**Завдання 4.** Порівняння:

- для класу Participant імплементувати інтерфейс Comparable та перевизначити метод compareTo для сортування учасників по імені.

- створити Comparator для порівняння учасників по віку.

- \*створити компаратор з пріорітетом використовуючи можливості Java 8 (спочатку порівняння по імені, а потім по віку).

- продемонструвати роботу порівнянь на прикладі сортування учасників команд.



**Висновок:** Я створив міні проект Game з використанням узагальнень, клонування та порівняння об'єктів.

**ЛАБОРАТОРНА РОБОТА № 4**

**Тема:** Класи String, StringBuffer та StringBuilder. Локалізація та інтернаціоналізація. Робота з датами.

**Мета роботи:** Робота з класами String, StringBuffer, StringBuilder та їх методами; практика використання локалізації та інтернаціоналізації; робота з датами.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Завдання 2.** Практика методів класу String:

- Напишіть метод, який приймає як параметр будь-який рядок, наприклад “I learn Java!!!”.

- Роздрукувати останній символ рядка.

- Перевірити, чи закінчується ваш рядок підрядком "!!!".

- Перевірити, чи починається ваш рядок підрядком "I learn ".

- Перевірити, чи містить ваш рядок підрядок "Java".

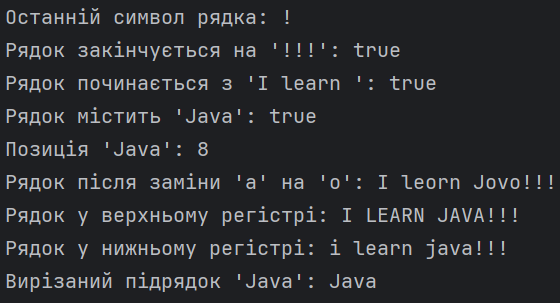
- Знайти позицію підрядка “Java” у рядку “I learn Java!!!”.

- Замінити всі символи "а" на "о".

- Перетворіть рядок на верхній регістр.

- Перетворіть рядок на нижній регістр.

- Вирізати рядок Java



**Завдання 3.** Створити рядок за допомогою класу StringBuilder або StringBuffer та його методів:

- Дано два числа, наприклад, 4 і 36, необхідно скласти наступні рядки: 4 + 36 = 40 4 - 36 = -32 4 \* 36 = 144

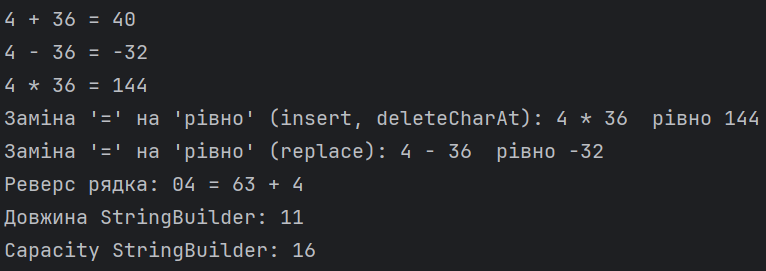
- Використати метод StringBuilder.append().

- Замініть символ “=” на слово “рівно”. Використати методи StringBuilder.insert(), StringBuilder.deleteCharAt().

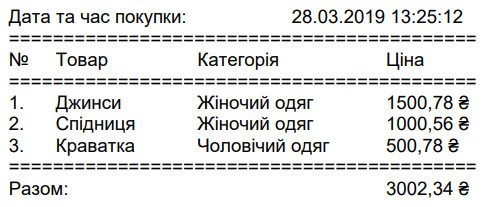
- Замініть символ “=” на слово “рівно”. Використати метод StringBuilder.replace().

- Змінити послідовність розташування символів в рядку на протилежну. Використати метод StringBuilder.reverse

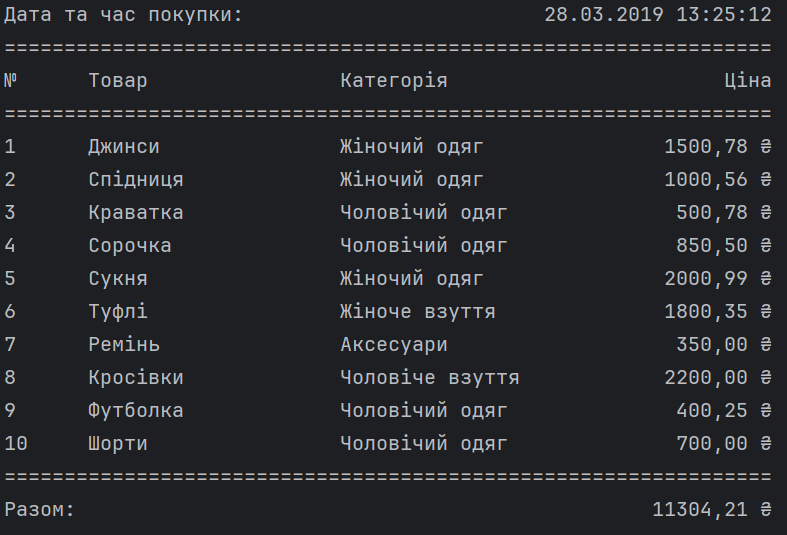
- Визначити довжину та capacity



**Завдання 4.** Вивести у форматованому вигляді чек з купленими товарами використовуючи можливості класу Formatter:



Доповнити список товарів до 10 шт.



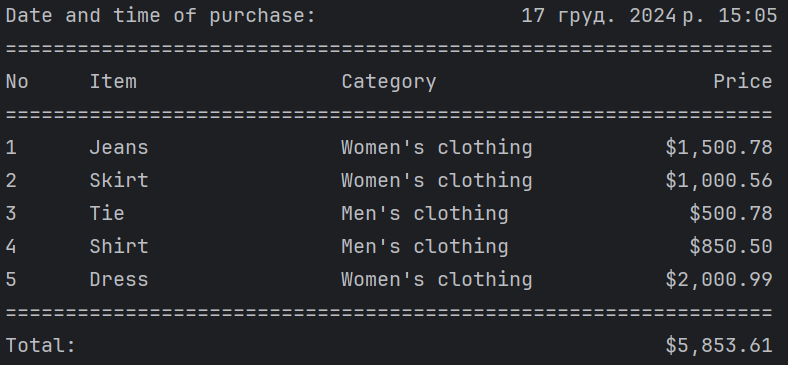
**Завдання 5.** Реалізувати інтернаціоналізацію для відображення чеку з товарами українською, англійською та будь-якою третьою мовою на ваш вибір. Для цього використати класи Locale та ResourceBundle. Для виведення валюти країни використати можливості класу NumberFormat.

- Створити директорію resources в корені проекту та позначити її як директорію з ресурсами.

- Створити три файли з розширенням properties для кожної локалі (наприклад: data\_ua\_UA) та заповнити даними (для кирилиці використати escape послідовності).

- Об'єднати їх у Resourse Bundle

- Реалізувати функціонал отримання та роботи з даними для кожної локалі.



**Завдання 6**. Робота з датами:

- Створіть об'єкт будь-якого класу для роботи з датами на власний вибір, вказуючи дату та час початку сьогоднішньої лабораторної з Java.

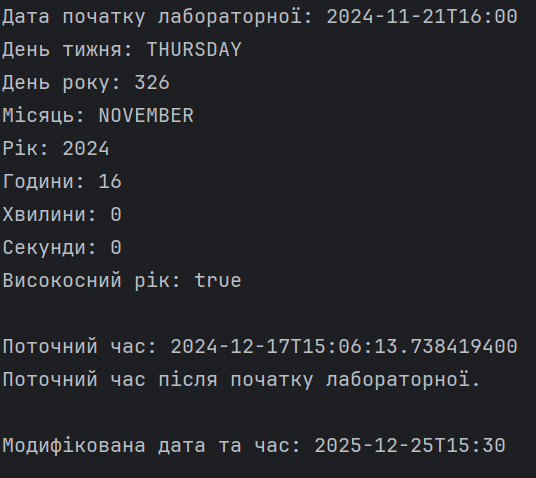
- Вивести на консоль день тижня, день у році, місяць, рік, години, хвилини, секунди.

- Перевірити чи рік високосний.

- Створіть об'єкт будь-якого класу для роботи з датами, який представляє поточний час.

- Порівняйте його з датою початку лабораторної з Java, використовуючи методи isAfter(), isBefore().

- Змініть значення елементів дати та часу на власний розсуд використовуючи методи обраного вами класу для роботи з датами.



**Висновок:** Я попрацював з класами String, StringBuffer, StringBuilder та їх методами; попрактикував використання локалізації та інтернаціоналізації; робота з датами.

**ЛАБОРАТОРНА РОБОТА № 5**

**Тема:** Java Collections Framework.

**Мета роботи:** Робота з Java Collections Framework.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

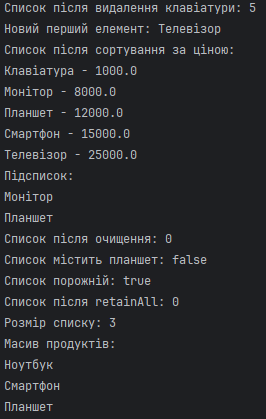
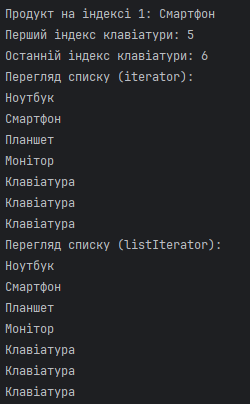
**Завдання 1.** Створити консольний Java проект java\_lab\_5 з пакетом com.education.ztu

**Завдання 2.** Створити клас Product та задати йому поля та методи на власний вибір.

**Завдання 3.** Створити динамічний масив, що містить об'єкти класу Product:

• Використовуємо клас ArrayList або LinkedList.

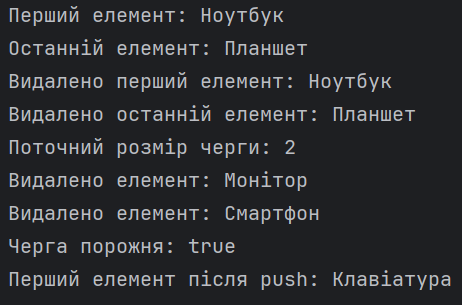
• Продемонструвати роботу з масивом використовуючи різні методи (add, addAll, get, indexOf, lastIndexOf, iterator, listIterator, remove, set, sort, subList, clear, contains, isEmpty, retainAll, size, toArray).



**Завдання 4.** Створити чергу, що містить об'єкти класу Product:

• Використовуємо клас ArrayDeque.

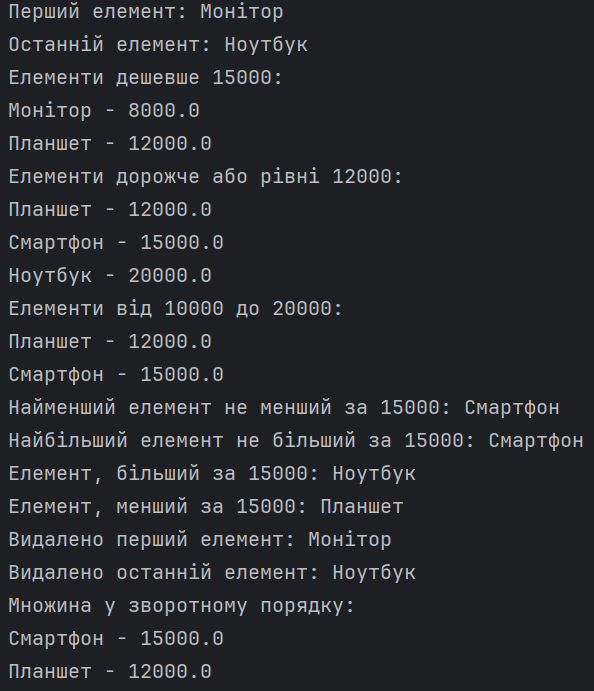
• Продемонструвати роботу з чергою використовуючи методи (push, offerLast, getFirst, peekLast, pop, removeLast, pollLast та інші).



**Завдання 5.** Створити множину, що містить об'єкти класу Product:

• Використовуємо клас TreeSet.

• Продемонструвати роботу з множиною використовуючи методи (add, first, last, headSet, subSet, tailSet, ceiling, floor, higher, lower, pollFirst, pollLast, descendingSet).

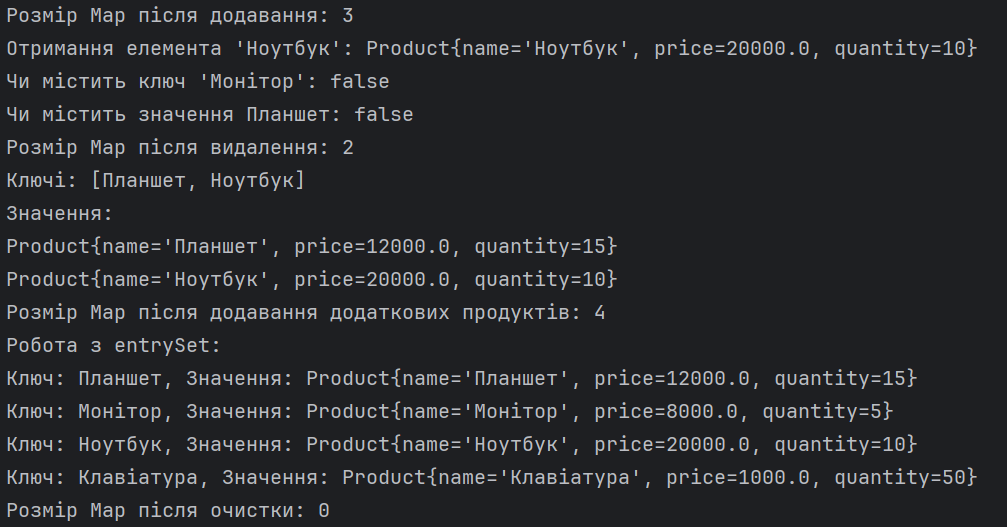


**Завдання 6.** Створити Map що містить пари (ключ, значення) - ім'я продукту та об'єкт продукту (клас Product).

• Використовуємо клас HashMap.

• Продемонструвати роботу з Map використовуючи методи (put, get, get, containsKey, containsValue, clear, putIfAbsent, keySet, values, putAll, remove, size) .

• Викликати метод entrySet та продемонструвати роботу з набором значень, що він поверне (getKey, getValue, setValue) .



**Завдання 7.** Продемонструвати роботу з класом Collections:

• Для роботи використати масив створений через Arrays.asList

• Метод Collections.sort()

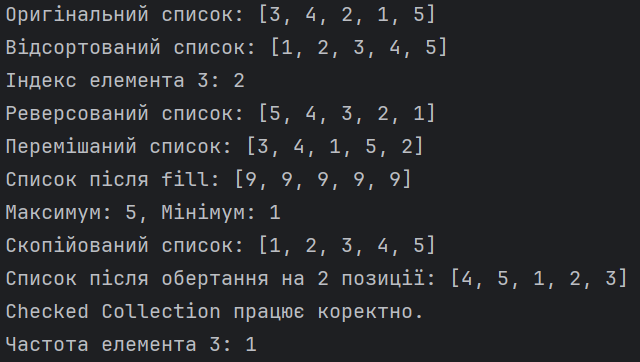
• Метод Collections.binarySearch()

• Методы Collections.reverse(), Collections.shuffle()

• Метод Collections.fill() • Методы Collections.max(), Collections.min()

• Метод Collections.copy() • Метод Collections.rotate()

• Метод Collections.checkedCollection() • Метод Collections.frequency()



**Висновок:** Я працював з Java Collections Framework.

**ЛАБОРАТОРНА РОБОТА № 6**

**Тема:** Обробка виключних ситуацій. Потоки вводу-виводу. Робота з файлами.

**Мета роботи:** Обробка виключних ситуація, створення власних класів винятків, робота з потоками вводу-виводу.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Завдання 1.** Створити консольний Java проект java\_lab\_6 з пакетом com.education.ztu. Створіть в корені проекту папку directory\_for\_files. Всі файли з якими ви будете працювати при виконанні завдань повинні знаходитись в ній.

**Завдання 2.** Перевірка логіну та паролю:

• Створити статичний метод checkCredentials, який приймає на вхід три параметри: login, password і confirmPassword.

• Login повинен містити лише латинські літери, цифри та знак підкреслення. Довжина login має бути меншою за 20 символів. Якщо login не відповідає цим вимогам, необхідно викинути WrongLoginException.

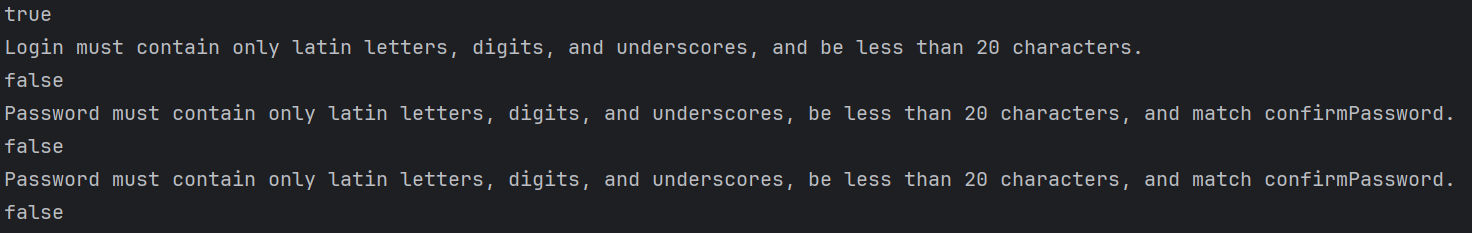
• Password повинен містити лише латинські літери, цифри та знак підкреслення. Довжина password має бути менше 20 символів. Також password і confirmPassword повинні бути рівними. Якщо password не відповідає цим вимогам, необхідно викинути WrongPasswordException.

• WrongPasswordException і WrongLoginException - користувацькі класи виключення з двома конструкторами - один за замовчуванням, другий приймає повідомлення виключення і передає його в конструктор класу Exception.

• Обробка винятків проводиться усередині методу.

• Використовуємо multi-catch block.

• Метод повертає true, якщо значення є вірними або false в іншому випадку



**Завдання 3.** Запис звіту про покупки в текстовий файл та читання з нього:

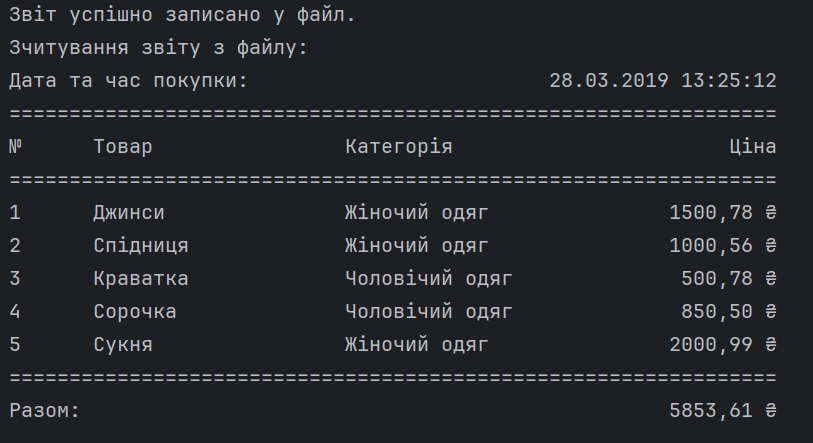
• Перевикористати код для формування звіту з покупок з лабораторної роботи 4

• Після покупки, записати звіт у файл, який містить інформацію про вміст кошика.

• Використовуємо клас FileWriter або PrintWriter для запису звіту.

• Використовуємо FileReader для читання звіту та відображення в консолі.

• Не використовувати try-with-resources.



**Завдання 4.** Копіювання файлу до іншого файлу:

• Написати клас, який копіює вміст текстового файлу та картинки з одного файлу до іншого.

• Використовуємо класи BufferedReader, FileReader, BufferedWriter, FileWriter, FileInputStream, FileOutputStream.

• Використати try-with-resources.

**Завдання 5.** Робота з класом RandomAccessFile:

• Дописати текст в декількох місцях в текстовому файлі. Можна використати текстовий файл зі списком товарів (наприклад, дописати декілька товарів) або будь-який інший файл з текстом.

**Завдання 6.** Робота з класом File:

• Створити нову папку з ім'ям inner\_directory.

• Вивести абсолютний шлях створеної папки.

• Вивести ім’я батьківської директорії.

• Створити два текстових файли всередині папки inner\_directory.

• Один файл видалити.

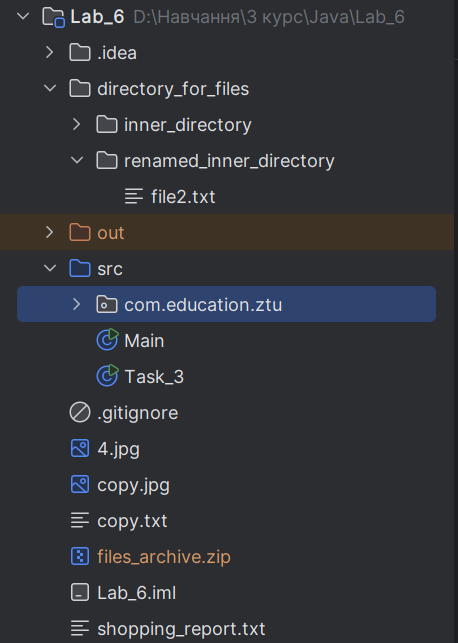
• Переіменувати папку inner\_directory в renamed\_inner\_directory

• Вивести список файлів та папок в папці directory\_for\_files, їх розмір та тип (файл, папка).

**Завдання 7.** Створення архіву:

• Додати всі створені файли в папці directory\_for\_files до архіву. Використати клас ZipOutputStream.

• Вивести список файлів з архіву. Використати клас ZipInputStream.



**Висновок:** Я обробив виключні ситуації, створив власні класи винятків, працював з потоками вводу-виводу.

**ЛАБОРАТОРНА РОБОТА № 7**

**Тема:** Багатопоточне програмування в Java.

**Мета роботи:** Практика роботи з потоками в Java.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Завдання 2.** Створити клас, що розширює Thread:

• Створити клас MyThread, що розширює Thread.

• Перевизначити метод run(). У циклі for вивести на консоль повідомлення «Я люблю програмувати!!!» 100 разів.

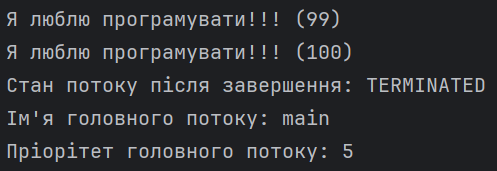
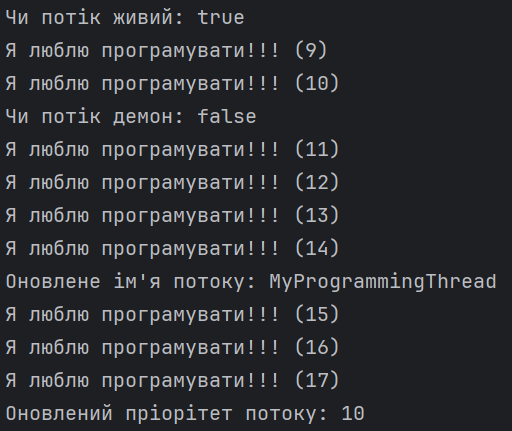
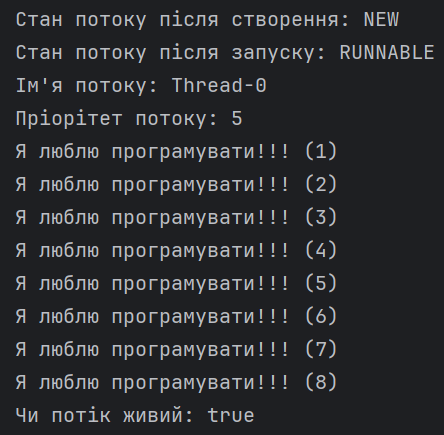
• Створити екземпляр класу та запустити новий потік.

• Вивести ім'я створеного потоку, його пріорітет, превірити чи він живий, чи є потоком демоном.

• Змінити ім'я, пріорітет створеного потоку та вивести в консоль оновлені значення.

• Після завершення роботи створеного потоку (використати метод join()) вивести ім’я головного потоку, та його пріорітет.

• Відобразити в консолі, коли ваш потік буде в стані NEW, RUNNUNG, TERMINATED



**Завдання 3.** Створити клас, що реалізує інтерфейс Runnable для виводу в консоль чисел від 0 до 10000, що діляться на 10 без залишку:

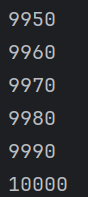
• Створити клас MyRunnable, який реалізує інтерфейс Runnable.

• Імплементувати метод run().

• Визначити умову, якщо потік хочуть перервати, то завершити роботу потоку та вивести повідомлення «Розрахунок завершено!!!»

• Створити три потоки, які виконують завдання друку значень.

• Використовуємо статичний метод Thread.sleep(), щоб зробити паузу на 2 секунди для головного потоку, а після цього викликати для створених потоків метод interrupt().



**Завдання 4.** Створити клас, що реалізує інтерфейс Runnable для вививедення арифметичної прогресії від 1 до 100 з кроком 1:

• Створити клас, який реалізує інтерфейс Runnable.

• Створити об'єкт зі статичною змінною result для збереження значення арифметичної прогресії.

• Перевизначити метод run(). Створити цикл for. У циклі виводимо через пробіл значення змінної result. Та додаємо наступне значення до змінної result та чекаємо 0,2 секунду.

• Забезпечити корректну роботу використовуючи синхронізований метод.

• Створити три потоки, які виконують завдання друку значень.



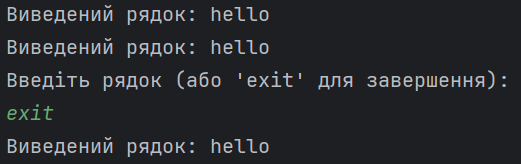
**Завдання 5.** Переробити 4 завдання використовуючи блок синхронізації.



**Завдання 6.** Створити два потоки Reader та Printer. Reader зчитує введені дані з консолі та записує в змінну. Після цього інформує потік Printer та засипає на 1 секунду, а потік Reader виводить дотриманий рядок. І так повторюється знову, поки користувач не завершить роботу програми.

• Змінну треба використати як об'єкт для синхронізації.

• Тут необхідно використати wait() i notify().



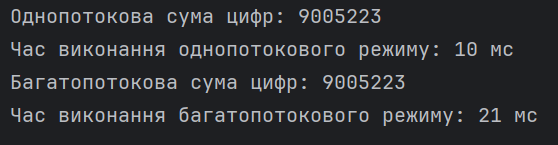
**Завдання 7.** Створити програму для знаходження суми цифр в масиві на 1 000 000 елементів:

• Заповнити масив числами використовуючи клас Random.

• Реалізувати задачу в однопоточному та багатопоточному середовищі.

• Для багатопоточного середовища використати ExecutorService на 5 потоків та об'єкти потоків, що імплементують інтерфейси Runnable або Callable.

• Заміряти час виконання обох варіантів завдання використовуючи System.currentTimeMillis() та вивести результати в консоль.



**Висновок:** Я практикував роботу з потоками в Java **ЛАБОРАТОРНА РОБОТА № 8**

**Тема:** Лямбда вирази. Функціональні інтерфейси. Посилання на методи. Stream API.

**Мета роботи:** Практика роботи з лямбда виразами, функціональними інтерфейсами; використання посилань на методи та Stream API при розробці програм на Java.

**Git:** https://github.com/ipz223-zdo/java\_labs

**Хід роботи**

**Завдання 2.** Описати власний функціональний інтерфейс Printable з методом void print() та написати лямбда вираз цього інтерфейсу.



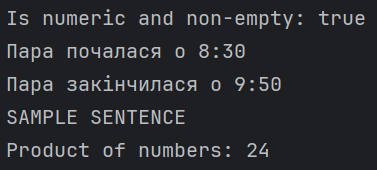
**Завдання 3.** Написати лямбда вирази для вбудованих функціональних інтерфейсів:

a) Створити лямбда вираз, який повертає значення true, якщо рядок можна привести до числа, використовуючи функціональний інтерфейс Predicate. Створити вираз лямбда, який перевіряє, що рядок можна привести до числа, використовуючи функціональний інтерфейс Predicate. Написати програму, яка перевіряє, що рядок можна привести до числа, використовуючи метод and() функціонального інтерфейсу Predicate.

b) Написати лямбда вираз, який приймає на вхід рядок і виводить на консоль повідомлення "Пара почалася о 8:30", "Пара закінчилася о 9:50". Використовуємо функціональний інтерфейс Consumer і метод за замовчуванням andThen.

c) Написати лямбда вираз, який виводить в консоль речення в з літерами у верхньому регістрі. Використовуємо функціональний інтерфейс Supplier.

d) Написати лямбда вираз, який приймає на вхід рядок з набором чисел через пробіл та повертає добуток цих чисел. Використовуємо функціональний інтерфейс Function.



**Завдання 4.** Stream API.

• Створити стрім з масиву Product з полями name, brand, price, count.

• Отримати всі бренди та вивести в консоль. (map)

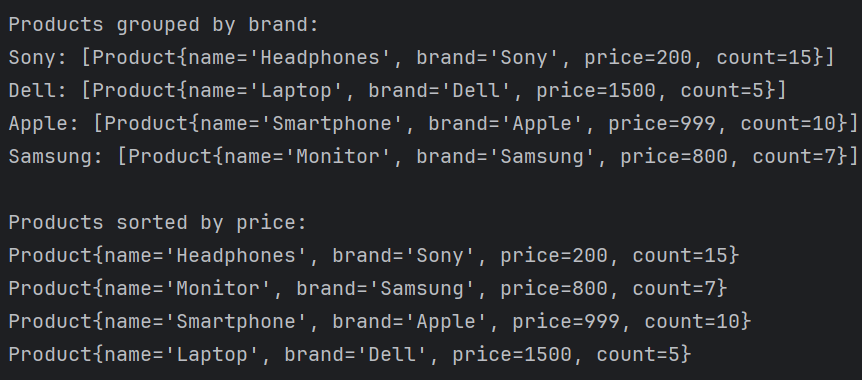
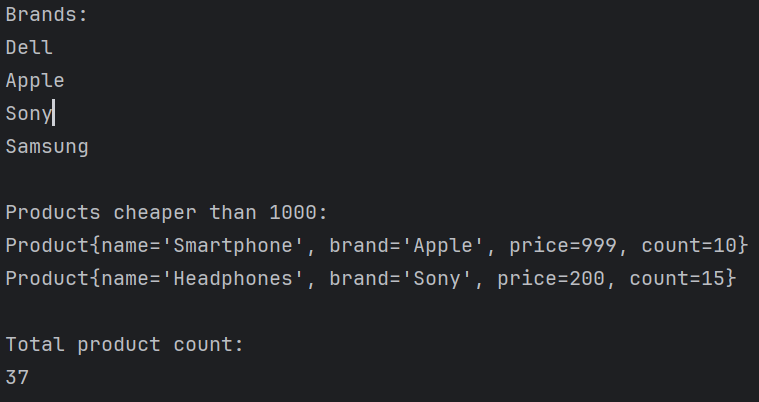
• Отримати 2 товари ціна яких менше тисячі. (filter, limit)

• Отримати суму всіх видів товарів, що є на складі. (reduce)

• Згрупувати товари по бренду (Collectors.groupingBy())

• Відсортувати товари за зростанням ціни та повернути масив (sorted, Collectors)

• За бажанням дописати функціонал, що використовує інші методи стрімів.



**Завдання 5.** Посилання на методи чи конструктори. В попередньому завданні, де це можливо, виклики переробити на посилання на методи чи конструктори.

**Завдання 6.** Використання Optional та його методів. Знайти максимальне значення з масиву чисел, в іншому випадку повернути рядок «Числа відсутні».



**Висновок:** Я попрактикувався з лямбда виразами, функціональними інтерфейсами, використав посилання на методи та Stream API при розробці програм на Java.