**Лабораторна робота №1**

**Типи даних, умови, цикли та масиви**

**Мета:** Використовуючи теоретичне підґрунтя про типи даних, що використовуються у мові Java, правила використання умовних переходів, написання циклів та використання масивів виконати дії що будуть вказано в завданні до лабораторної роботи.

# ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ

Встановити JDK на власному персональному комп’ютері та будь-яку IDE. Також допускається виконання лабораторних робіт у будь-якому іншому текстовому редакторі.

Створити акаунт на github у випадку відсутності.

Створити репозиторій що має назву яка задовольняє наступному шаблону **OOP-KB-22{1 or 2}-Name-Surname** (наприклад OOP-KB223-Ihor-Diuba). Результати роботи необхідно буде завантажити на github для перевірки.

При використанні IDE створити проект під назвою **lab01** для розміщення результатів виконання завдань. У випадку використання звичайного текстового редактору створити окрему директорію **lab01** для виконання завдань.

Виконайте наступні завдання(кожне завдання має бути виконано в окремому файлі).

**ЗМІСТ**

[1. ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ 1](#_Toc163412820)

[2. ВИКОНАННЯ ЗАВДАНЬ ЛАБОРАТОРНОЇ РОБОТИ 3](#_Toc163412821)

[2.1. Завдання 1 3](#_Toc163412822)

[2.2. Завдання 2 3](#_Toc163412823)

[2.3. Завдання 3 6](#_Toc163412824)

[2.4. Завдання 4 7](#_Toc163412825)

[2.5. Завдання 5 8](#_Toc163412826)

[2.6. Завдання 6 11](#_Toc163412827)

[2.7. Завдання 7 12](#_Toc163412828)

[2.8. Завдання 8 13](#_Toc163412829)

[2.9. Завдання 9 15](#_Toc163412830)

[2.10. Завдання 10 16](#_Toc163412831)

[2.11. Завдання 11 18](#_Toc163412832)

[2.12. Завдання 12 19](#_Toc163412833)

[2.13. Завдання 13 21](#_Toc163412834)

[2.14. Завдання 14 24](#_Toc163412835)

[3. ВИСНОВОК 27](#_Toc163412836)

# ВИКОНАННЯ ЗАВДАНЬ ЛАБОРАТОРНОЇ РОБОТИ

## Завдання 1

|  |  |
| --- | --- |
| 1 | Напишіть програму, яка прочитає рядок із System.in і виведе повідомлення: "Hello, прочитаний рядок". Якщо у вхідному рядку кілька слів, потрібно надрукувати їх усі. |

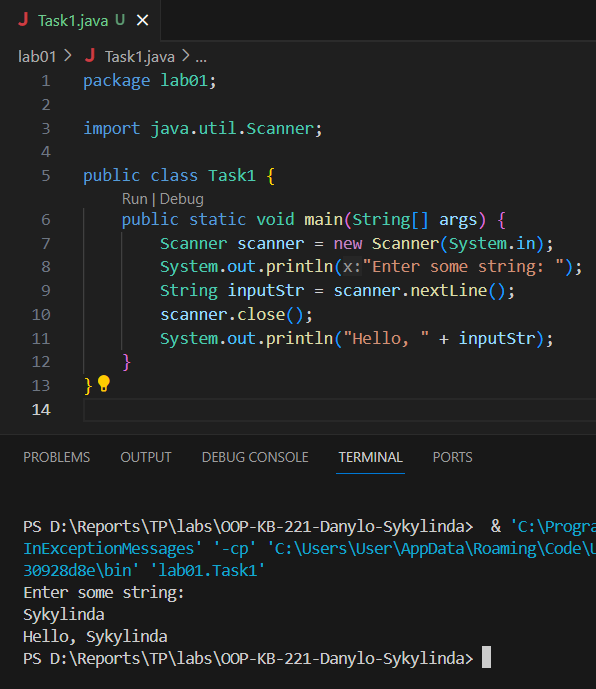


Рис. 1. Перше завдання.

## Завдання 2

|  |  |
| --- | --- |
| 2 | Напишіть програму, яка отримує на вхід ціле число. Це число є поточний добовий час, який обчислюється як кількість секунд, що пройшли з півночі. Потім програма повинна надрукувати значення екрану електронного годинника для цього часу.  Вхідне значення отримується через System.in. Вихідне значення має бути надруковане у System.out.  Гарантується, що вхідне значення — невід'ємне.  Формат виводу: h:mm:ss (можливі значення: [0:00:00; 23:59:59]).  **Приклади**:  Введення даних: 60  Результат виконання: 0:01:00  Введення даних: 3599  Результат виконання: 0:59:59  Введення даних: 86229  Результат виконання: 23:57:09 |

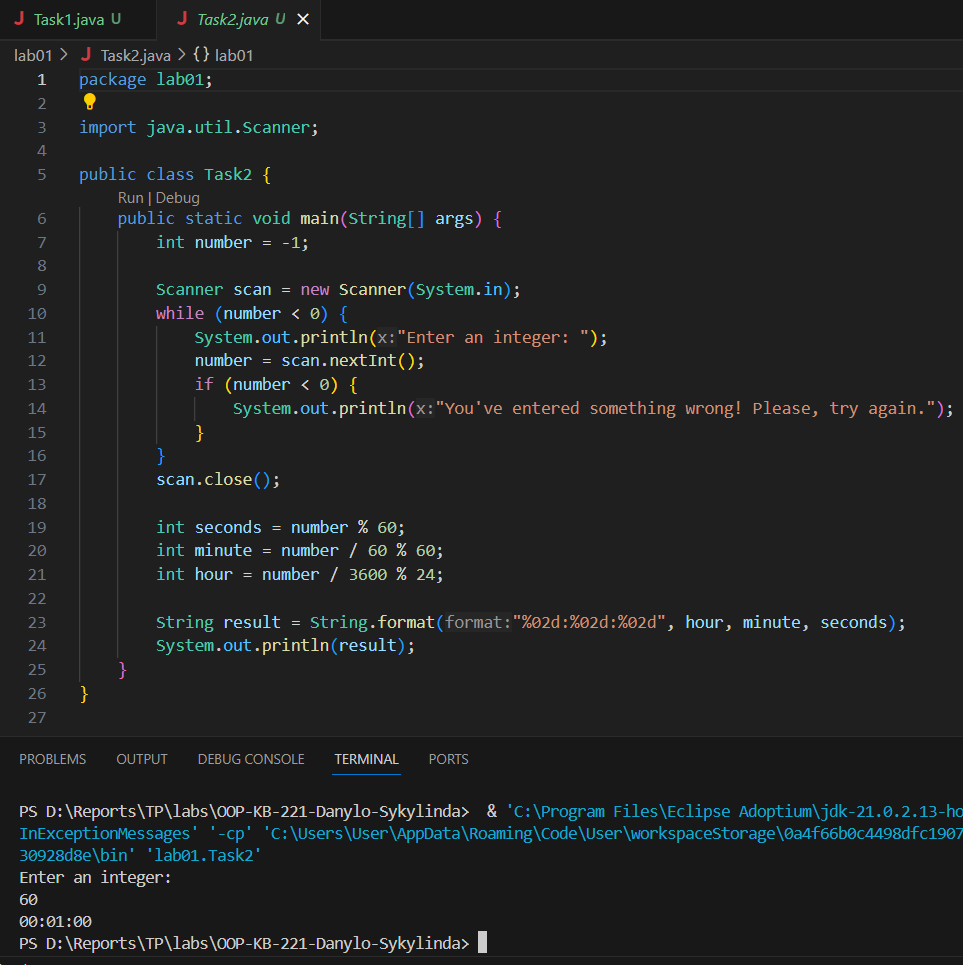


Рис. 2. Друге завдання.

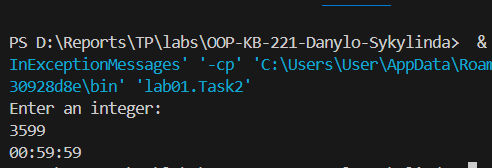


Рис. 3. Продовження другого завдання. Перевірка роботи коду.

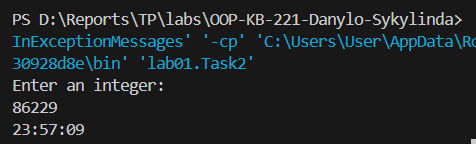


Рис. 4. Продовження другого завдання. Перевірка роботи коду.

## Завдання 3

|  |  |
| --- | --- |
| 3 | Напишіть програму, яка запитує на вхід число-пароль. Якщо воно дорівнює секретній кількості, програма виводить: "Hello, Agent", інакше друкує: "Access denied".  Секретний пароль зберігається в final static int password. |

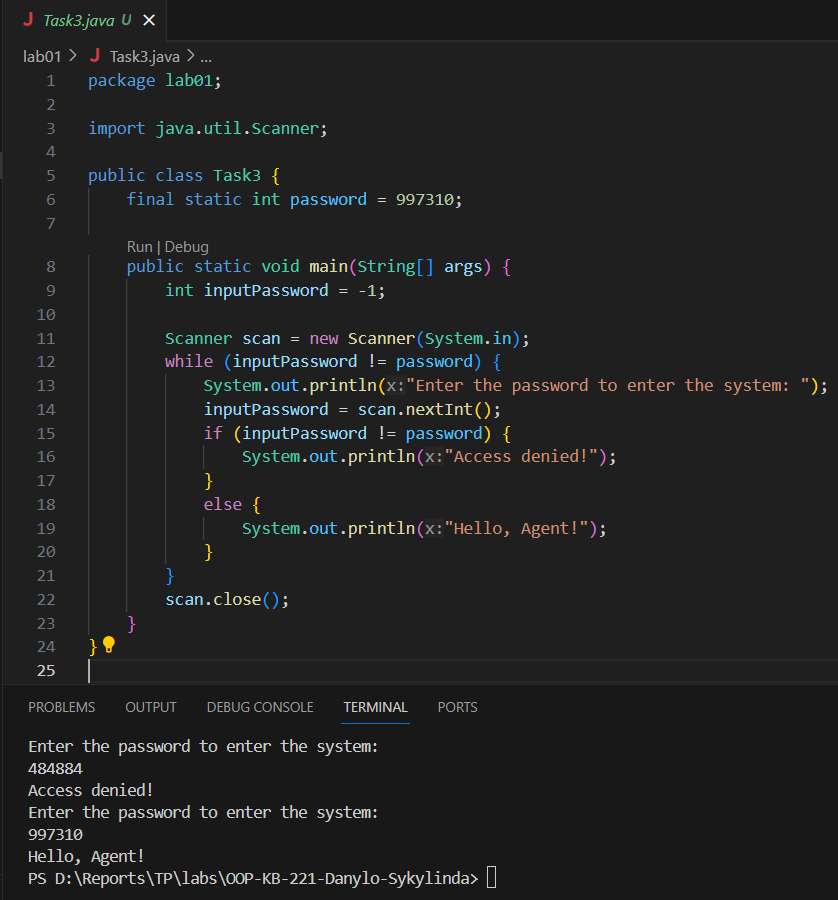


Рис. 5. Третє завдання.

## Завдання 4

|  |  |
| --- | --- |
| 4 | Напишіть просту програму, яка запитує кількість незнайомців, з якими потрібно зустрітися, рядково читає імена незнайомців, по рядках виводить: "Hello, (ім'я незнайомця)" для кожного незнайомця.  Доповнити програму перевірками на кількість, що є від’ємною, та рівна 0. |

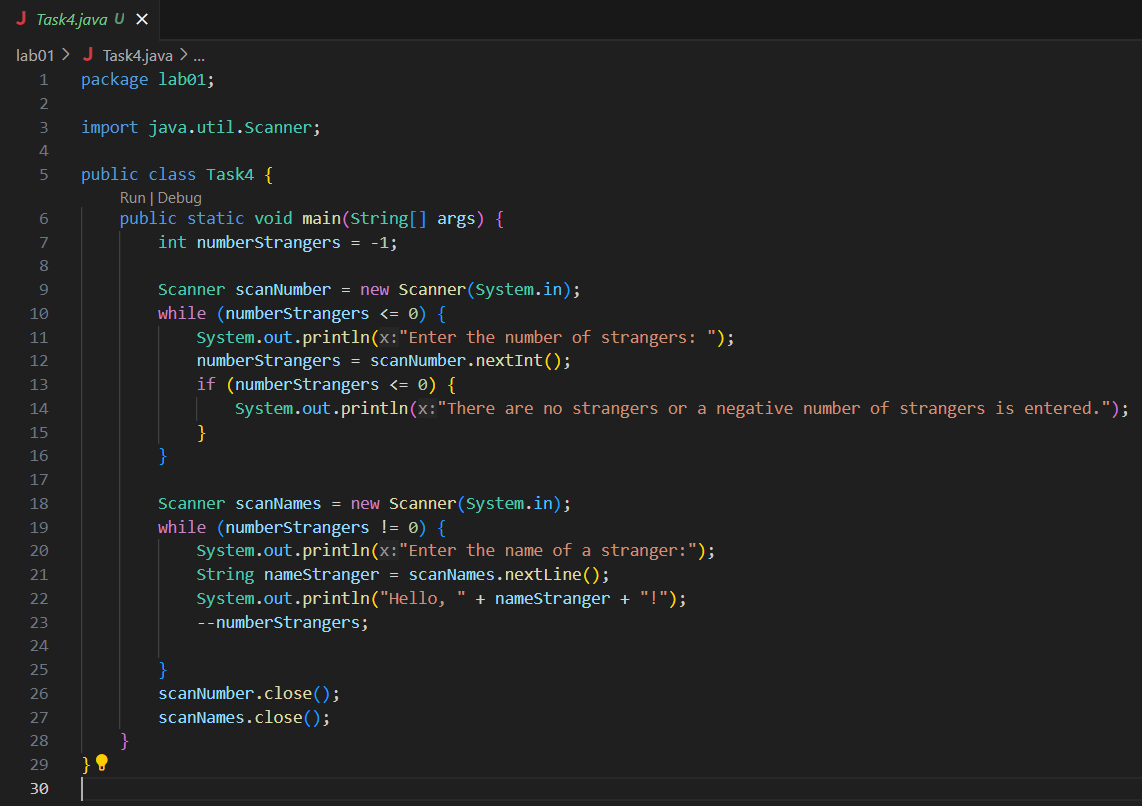


Рис. 6. Четверте завдання.

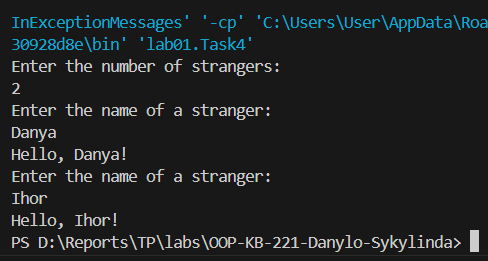


Рис. 7. Перевірка коду четвертого завдання.

## Завдання 5

|  |  |
| --- | --- |
| 5 | Наведено таку умову: равлик піднімається по дереву на a футів на день. Потім щоночі равлик сповзає вниз на b футів. Висота дерева — h футів.  Напишіть програму з підрахунком кількості днів, які знадобляться равлику, щоб дістатися до вершини дерева.  Програма читає a, b, h рядково. Вхідні значення гарантовано є додатніми цілими числами.  Якщо равлик не може дістатися до вершини дерева, має виводитися повідомлення: Impossible  **Приклади**:  Введення даних: 4 2 14  Результат: 6  Введення даних: 4 3 10  Результат: 7  Введення даних: 4 4 10  Результат: Impossible |

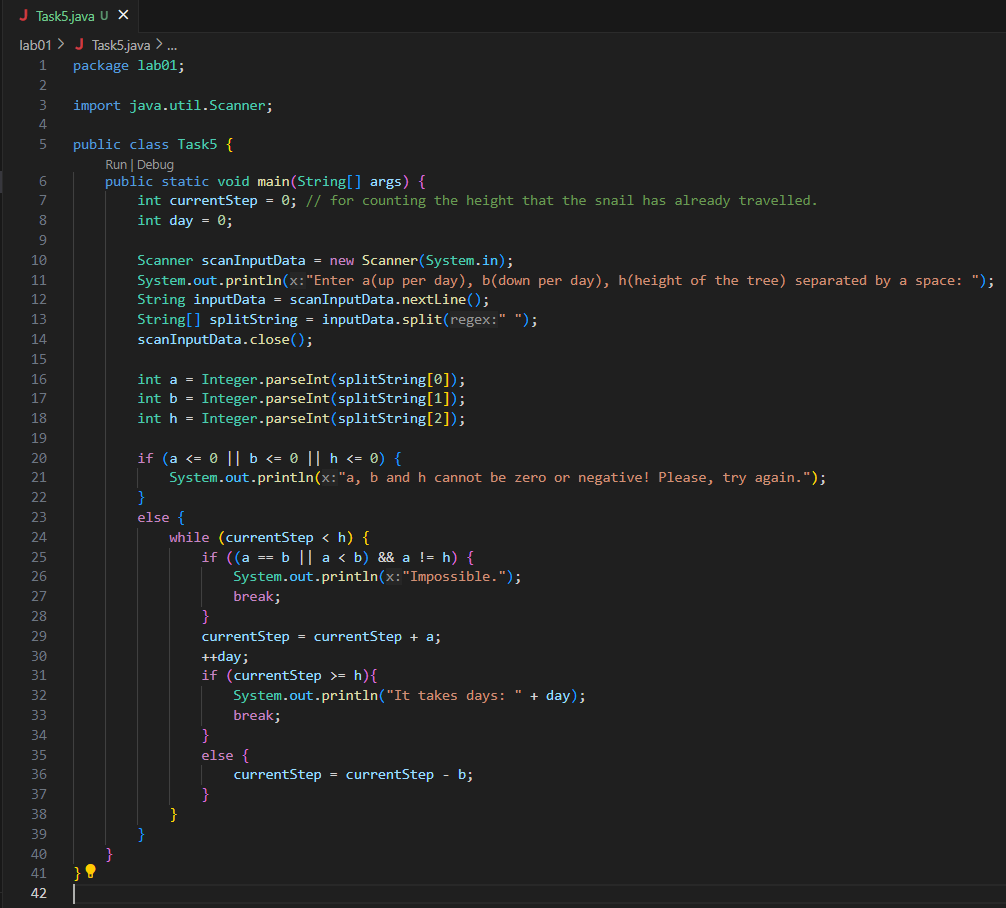


Рис. 8. П’яте завдання.

Викликаємо зчитувач, зчитуємо стрічку за допомогою nextLine(), далі отриману строку за допомогою методу split() розділяємо на окремі елементи, які в сукупності будуть масивом splitString. Закриваємо зчитувач. Далі зпарсуємо цілі значення, які отримали для розрахунку(кількість кроків вперед за день, кількість кроків назад вночі, висота дерева) із масиву елементів розділеної строки за допомогою методу parseInt() з класу Integer.

Робимо перевірку, що ці отримані значення від користувача не є від’ємними та нулем. Якщо не є, то далі створюємо цикл while з умовою допоки currentStep(поточна відстань, яку пройшов равлик по дереву) менше ніж висота дерева, то якщо «а»(крок вверх) дорівнює «b»(крок вниз) і «а» не дорівнює «h», то виводимо, що неможливо(Impossible) і припиняємо цикл. Навіщо потрібна умова, що «а» не дорівнює «h», оскільки «а» може бути, наприклад, 3 і висота дерева(h) 3, то виходить, що він за день дійде до кінця дерева і неважливо, який буде крок b.

Далі, якщо на умові цикл не зупинився, то додаємо до currentStep «а» - тобто, це відстань, яку він пройшов за день, а також інкрементуємо змінну day.

Далі робимо умову, що якщо поточний крок(currentStep) більше або дорівнює висоті дерева(h), то виводимо повідомлення про те, скільки днів равлику знадобилося, щоб подолати всю висоту дерева і перериваємо цикл. Якщо ж ця умова не виконується, то зменшуємо currentStep на b(кількість кроків вниз вночі).

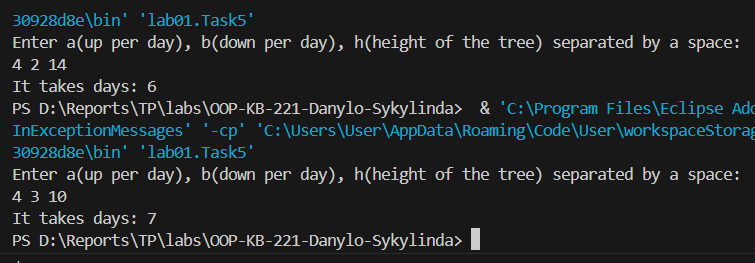


Рис. 9. Продовження п’ятого завдання. Перевірка коду завдання.

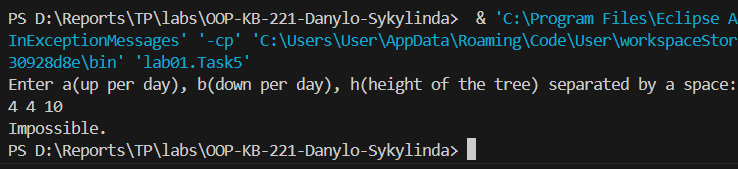


Рис. 10. Продовження п’ятого завдання. Перевірка коду завдання.

## Завдання 6

|  |  |
| --- | --- |
| 6 | Дано така умова: компанія друзів відвідує ресторан. Вони вирішили розділити рахунок порівну та додати 10 відсотків від загальної суми рахунку як чайові. Далі друзі покривають рівними частинами загальну суму платежу.  Напишіть програму, яка зчитує загальну суму рахунку та кількість друзів, а потім виводить розмір частини оплати.  Доповнити програму перевірками на від’ємну суму рахунку та нульову кількість друзів.  **Приклад**:  Введення даних: 1000 5  Результат: 220 |

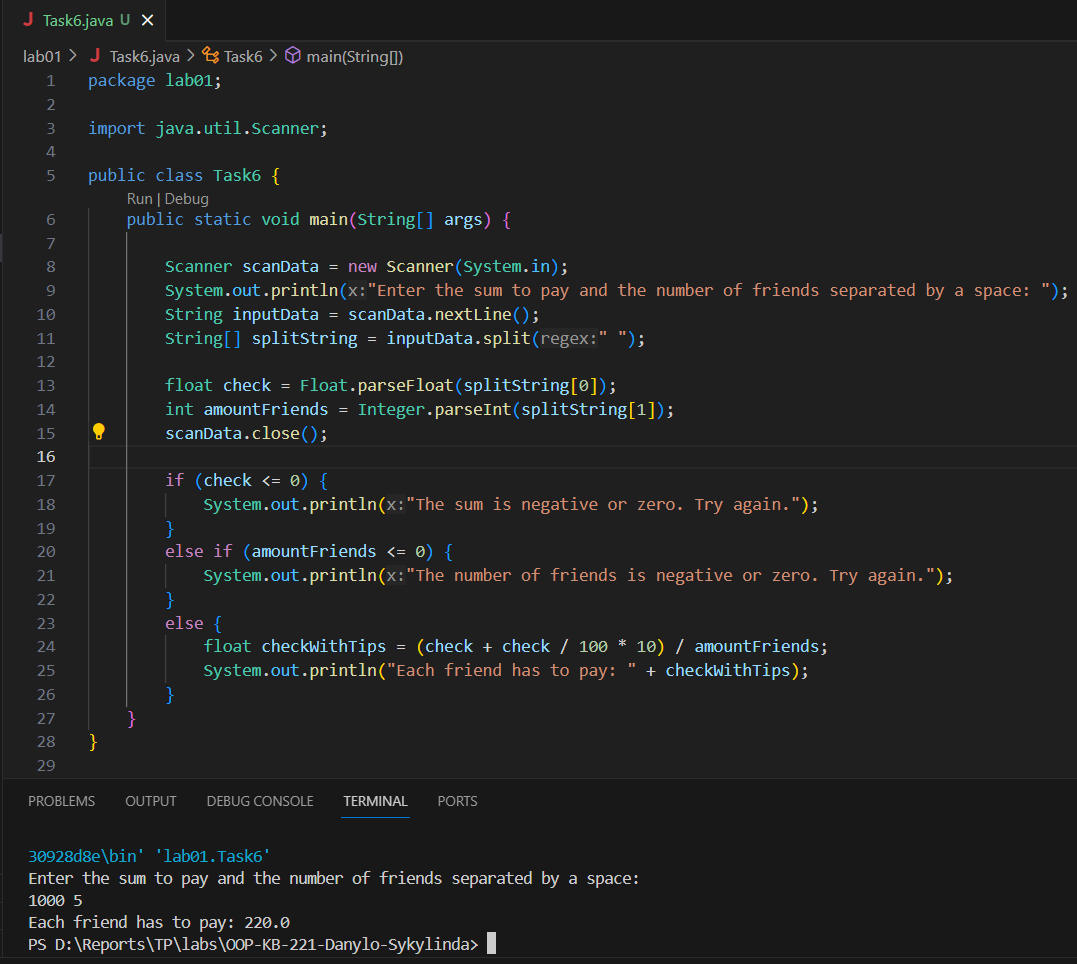


Рис. 11. Шосте завдання.

Практично все аналогічно, як було у п’ятому завданні.

## Завдання 7

|  |  |
| --- | --- |
| 7 | Напишіть програму, яка зчитує послідовність цілих значень зі стандартного введення і знаходить максимальне з них. Ви повинні зчитувати значення послідовності, доки наступне не стане 0. Нульове значення означає кінець вхідної послідовності та не є її елементом. Послідовність гарантовано містить хоча б одне значення.  **Приклад**:  Введення даних: 2 4 6 9 2 4 5 0  Результат: 9 |

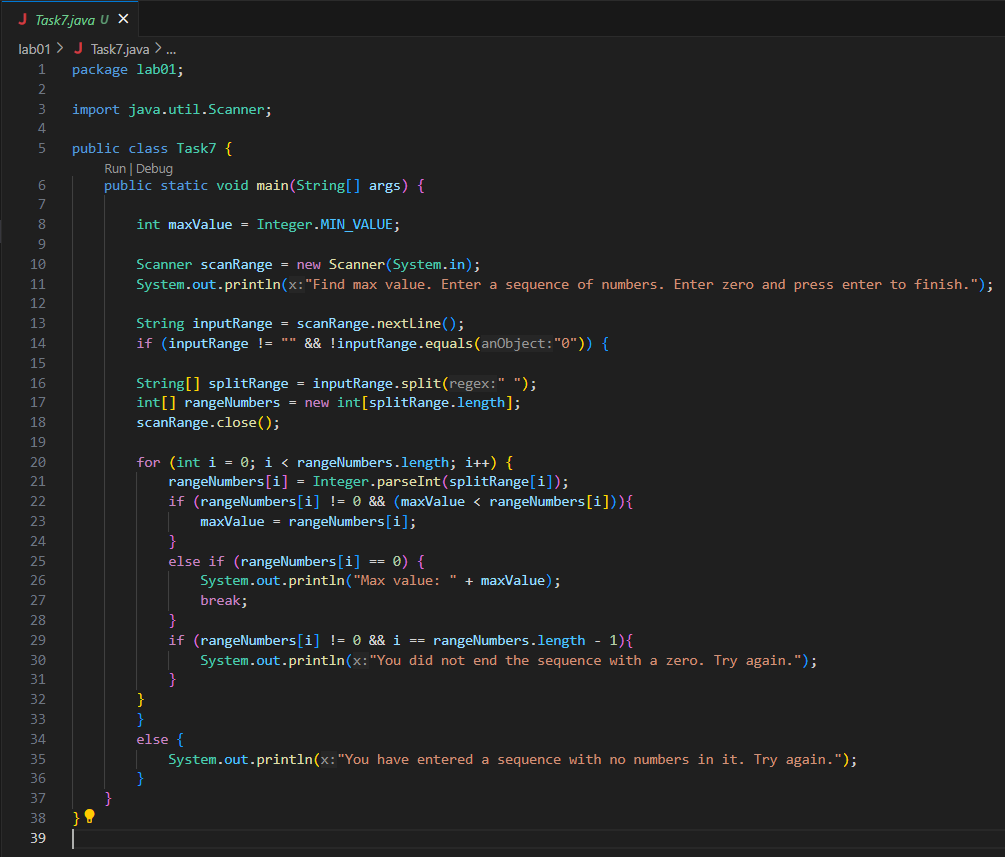


Рис. 12. Сьоме завдання.

Оголосили змінну maxValue і призначили їй найменше можливе ціле значення. Далі викликали зчитувач і почали зчитувати строку(послідовність).

Потім пишемо умову, якщо строка не є пустою і не закінчується нулем, то розділяємо зчитану строку(inputRange) на окремі елементи, що в сукупності будуть масивом значень. Далі оголошуємо масив rangeNumbers з величиною як масив розділеної строки(splitRange), у якому будемо зберігати цілі значеня. Закриваємо сканер. Далі цикл for допоки «i» не більше довжини масиву rangeNumbers, у ньому при кожній ітерацій для кожного елемента rangeNumbers присвоюємо зпарсене значення з масиву розділеної строки(splitRange), далі іде умова, якщо значення поточного елементу(rangeNumbers[i]) не дорівнює нулю і максимальне значення менше ніж значення поточного елементу, то змінній максимального значення(maxValue) присвоюємо значення поточного елементу(rangeNumbers[i]).

Інакше якщо значення поточного елементу дорівнює нуль, то виводимо максимальне число із послідовності.

Прописуємо ще одну умову, що якщо rangeNumbers[i] != 0 та i == rangeNumbers.length – 1, то виводимо повідомлення проте, що користувач забув завершити послідовність нулем, тобто, ми перевіряємо останній елемент послідовності.

І повертаючись до першої умови, де ми перевіряли, чи увів щось взагалі користувач, чи увів нуль, то якщо воно не виконується, то пишемо, що у послідовності немає жодного числа.

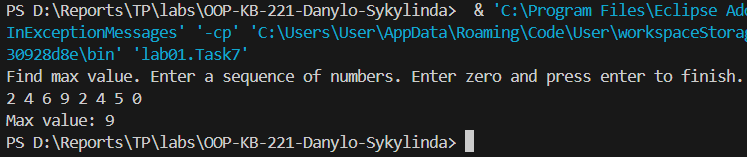


Рис. 13. Продовження сьомого завдання. Перевірка роботи коду завдання.

## Завдання 8

|  |  |
| --- | --- |
| 8 | Напишіть програму, яка зчитує послідовність цілих значень зі стандартного введення та знаходить середнє значення. Ви повинні зчитувати значення послідовності, доки наступне не стане 0. Нульове значення означає кінець вхідної послідовності та не є її частиною. Послідовність гарантовано містить хоча б одне значення.  **Приклад:**  Введення даних: 1 2 3 4 5 0  Результат: 3 |

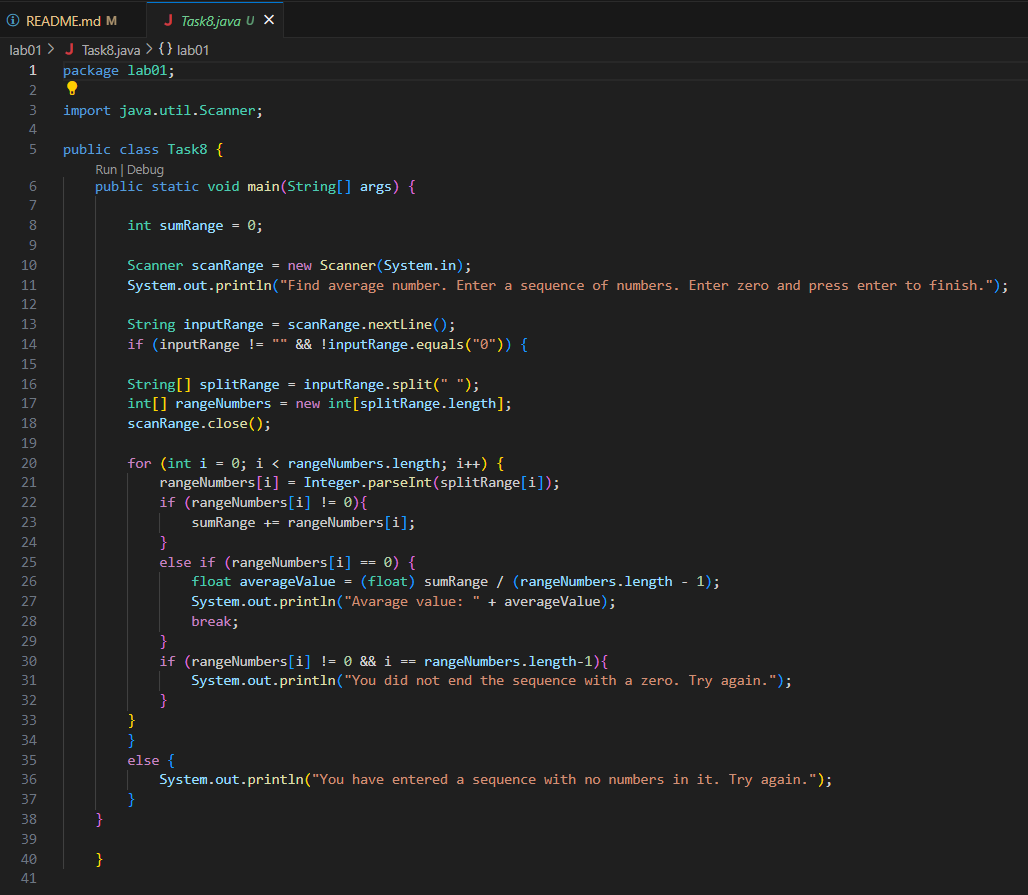


Рис. 14. Восьме завдання.

Принцип виконання завдання практично однаковий щодо перевірок та розділення строки на елементи, та перетворення елементів типу строки в елементи типу цілих значень, як і в сьомому завданні. Лише суть завдання різниться. Роблю перевірку, що rangeNumbers[i] == 0, якщо це правда, то сумую всі елементи послідовності та ділю на кількість елементів мінус один(мінус один, оскільки 0 не вважаємо елементом послідовності), і виводимо на екран, закінчуємо цикл. Все далі аналогічно як в сьомому завданні.

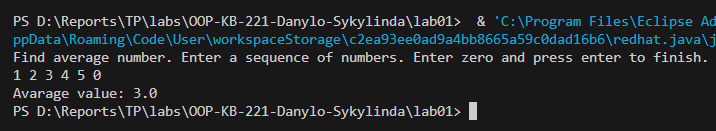


Рис. 15. Продовження сьомого завдання. Перевірка роботи коду завдання.

## Завдання 9

|  |  |
| --- | --- |
| 9 | Напишіть програму для вирішення квадратних рівнянь. При виконанні врахуйте залежність значення дискримінанту до кількості результатів  Приклад:  Введення даних: 2 5 -3  Результат: -3 0.5  Введення даних: 2 2 2  Результат : no roots |

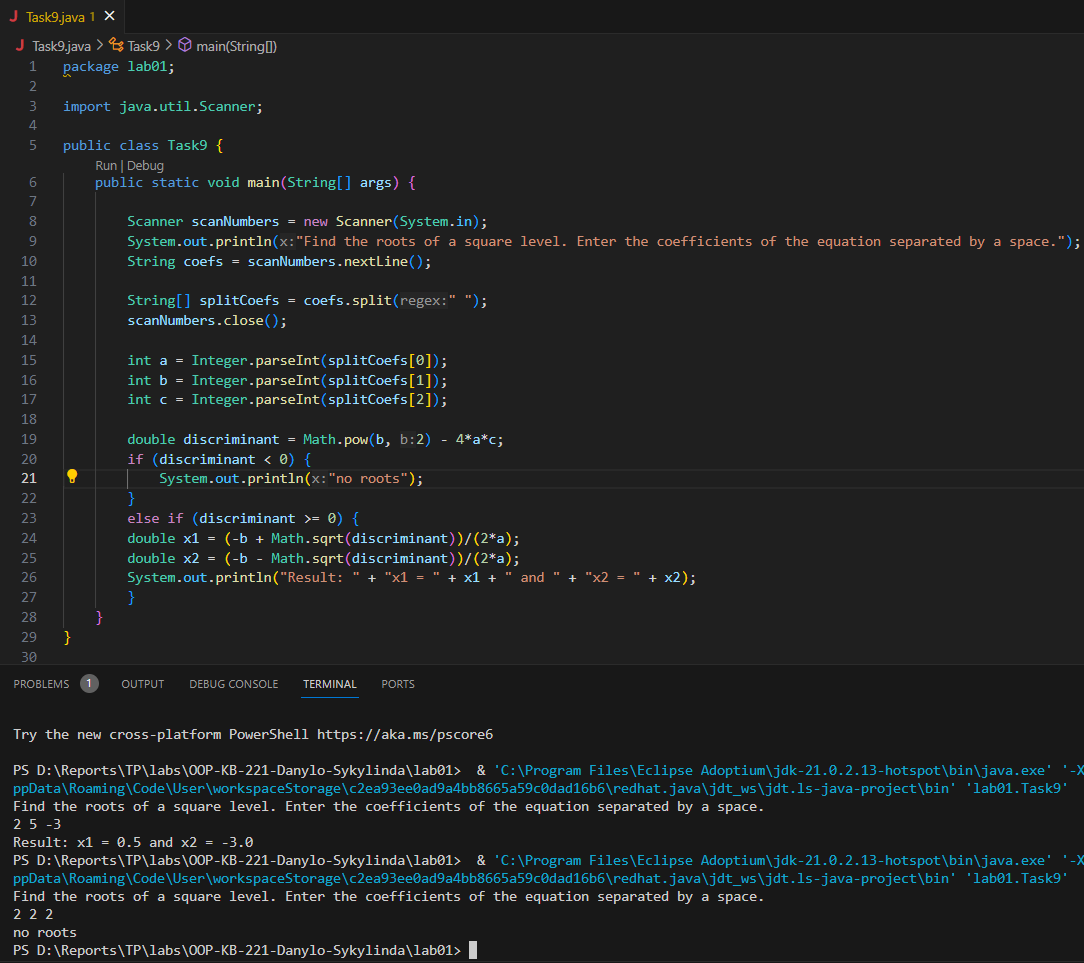


Рис. 16. Дев’яте завдання. Нижче на фото перевірка роботи коду.

Методи добуття необхідних значень зі строки, що увів користувач, є подібними до методів в попередньому завданні. Далі просто реалізується алгоритм пошуку дискримінанту квадратного рівняння, перевірка на невід’ємність дискримінанту і далі алгоритм пошуку коренів квадратного рівняння.

## Завдання 10

|  |  |
| --- | --- |
| 10 | Напишіть програму, що містить метод max(…). Правильна реалізація має отримати масив значень типу int та повернути його максимальне значення. Вхідний масив гарантовано не буде порожнім або null. Метод max не повинен змінювати масив. |

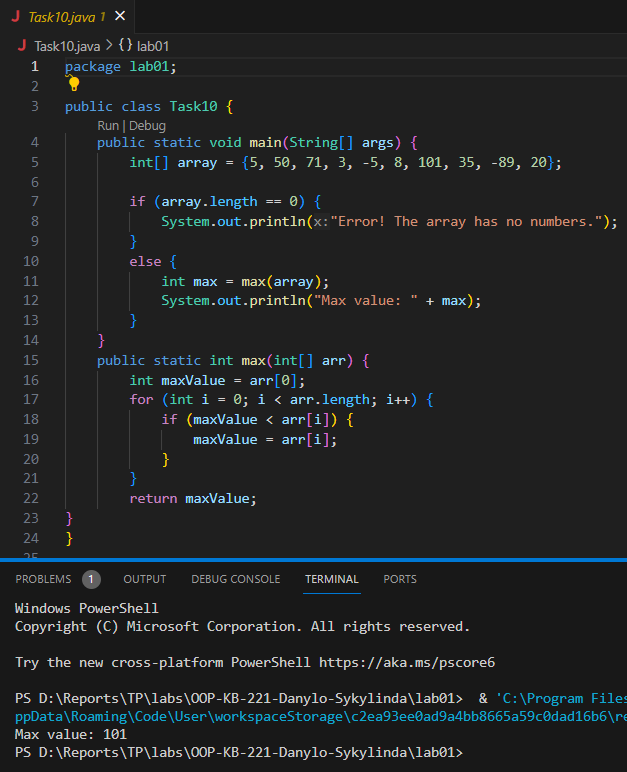


Рис. 17. Деcяте завдання. Нижче на фото перевірка роботи коду.

В цьому завданні я реалізував спочатку перевірку на пустоту масиву, якщо він не пустий, то створюю змінну int max, якій присвоюю виклик методу max(), передавши в якості аргументу масив array.

У самому методі max() реалізуємо такий функціонал, спочатку створюємо змінну int maxValue, яка буде слугувати для поточного максимального значення, і призначаємо їй значення першого елемента масиву arr(який є нашим аргументом). Далі реалізуємо цикл for, в якому будемо перебирати елементи масиву і порівнювати з поточним максимальним значенням(maxValue), якщо поточне максимальне значення менше, ніж елемент масиву arr з індексом i, то змінній maxValue присвоїти значення елементу масиву arr з індексом і. Не забудемо повернути змінну maxValue як результат в нашому методі.

## Завдання 11

|  |  |
| --- | --- |
| 11 | Напишіть програму, що містить метод sum. Правильна реалізація має отримати масив значень int та повернути суму парних чисел. Якщо цей масив дорівнює нулю або порожній, метод повертає 0. Метод sum не повинен змінювати масив. |

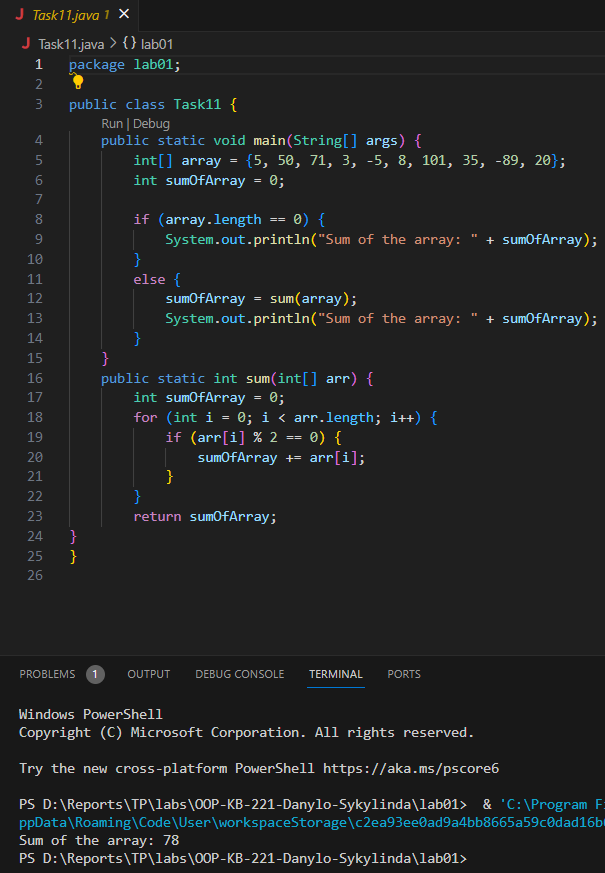


Рис. 18. Одинадцяте завдання. Нижче на фото перевірка роботи коду.

У цьому завданні я реалізував спочатку перевірку на пустоту масиву, якщо масив не пустий, то змінній sumOfArray я присвоюю виклик методу sum() і передаю йому масив array у якості аргументу. У самому методі sum() я створюю цикл, у якому я перебираю елементи масиву з умовою, що якщо діляться на 2 націло, то додаю значення поточного елементу(arr[i]) до змінної sumOfArray, яка змінною для обрахунку суми. Не забудемо повернути змінну sumOfArray.

## Завдання 12

|  |  |
| --- | --- |
| 12 | Напишіть програму, що містить метод getSumCheckArray. Правильна реалізація повинна отримати масив значень int і повернути масив логічних значень, де кожен елемент є результатом перевірки: чи є вихідний елемент сумою двох попередніх елементів у даному масиві.  Гарантується, що довжина масиву становить 2 або більше.  Цей масив гарантовано не дорівнює null.  Метод повертає масив логічних значень, де кожен елемент є результатом для відповідного елемента в даному масиві.  Перші два елементи логічного масиву завжди false.  Приклад  Введення даних: [1, -1, 0, 4, 6, 10, 15, 25]  Результат: [false, false, true, false, false, true, false, true] |

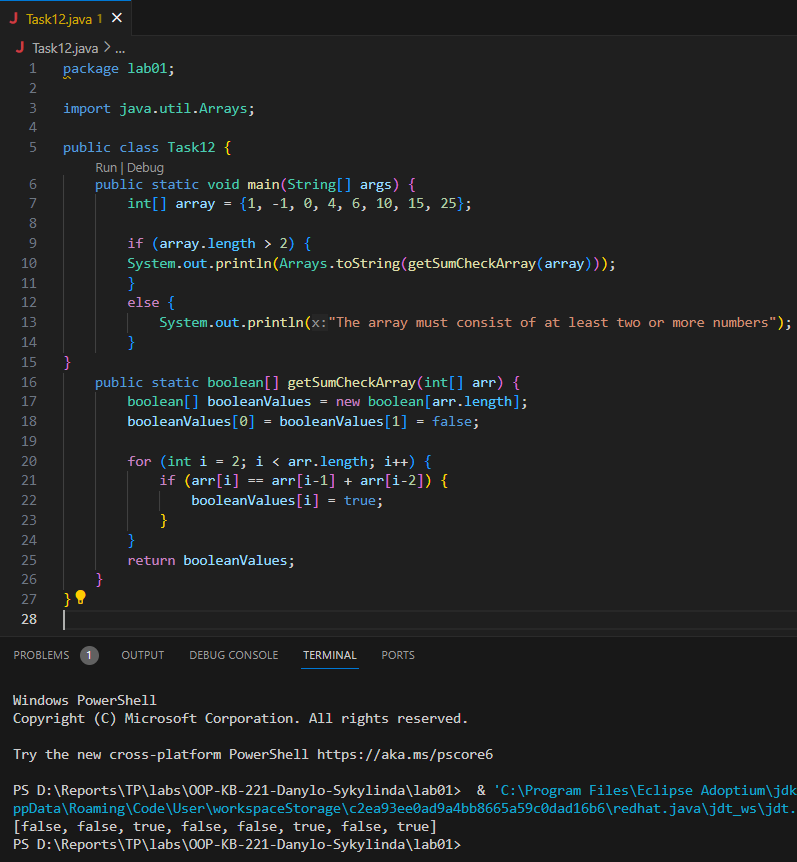


Рис. 19. Дванадцяте завдання. Нижче на фото перевірка роботи коду.

Спочатку перевірив довжину масиву, що вона більше двох, якщо вона більше двох, то виводимо виклик методу getSumCheckArray(), передавши цьому методу в якості аргументу масив array, і результат виклику перетворюємо в строку.

У самому методі getSumCheckArray() створюємо масив булевих значень booleanValues, який матиме розмір, як аргумент масив arr. За умовою задачі перші два елементи масиву booleanValues є false завжди, але це логічно, оскільки вони не мають двох елементів перед ними. Призначаємо елементам, що знаходяться по індексах 0 та 1 значення false. Далі я створив цикл for, який починаючи з третього елементу перебиратиме елементи масиву arr з умовою, що якщо поточний елемент, який ми зараз перебираємо(arr[i]) дорівнює сумі двох попередніх елементів (arr[i-1] та arr[i-2]), то присвоювати елементу масиву booleanValues з індексом «i» значення true. Не забудемо повернути масив booleanValues.

## Завдання 13

|  |  |
| --- | --- |
| 13 | Напишіть програму, що містить метод removeLocalMaxima(…). Правильна реалізація має отримати масив значень типу int і повернути копію даного масиву з усіма видаленими локальними максимумами. Вихідний масив не можна змінювати.  **Локальний максимум** — це елемент, який більший за будь-який з його сусідніх елементів. Вам необхідно видалити елементи, які є локальними максимумами у вихідному масиві.   * Розмір цього масиву гарантовано більше 1. * Розмір цього масиву гарантовано більше 1. * Якщо масив не має локальних максимумів, ви повинні повернути його копію без змін. * Ви можете використовувати методи java.util.Arrays.\*.   Приклад:  Введення даних: [18, 1, 3, 6, 7, -5]  Результат виконання: [1, 3, 6, -5] |

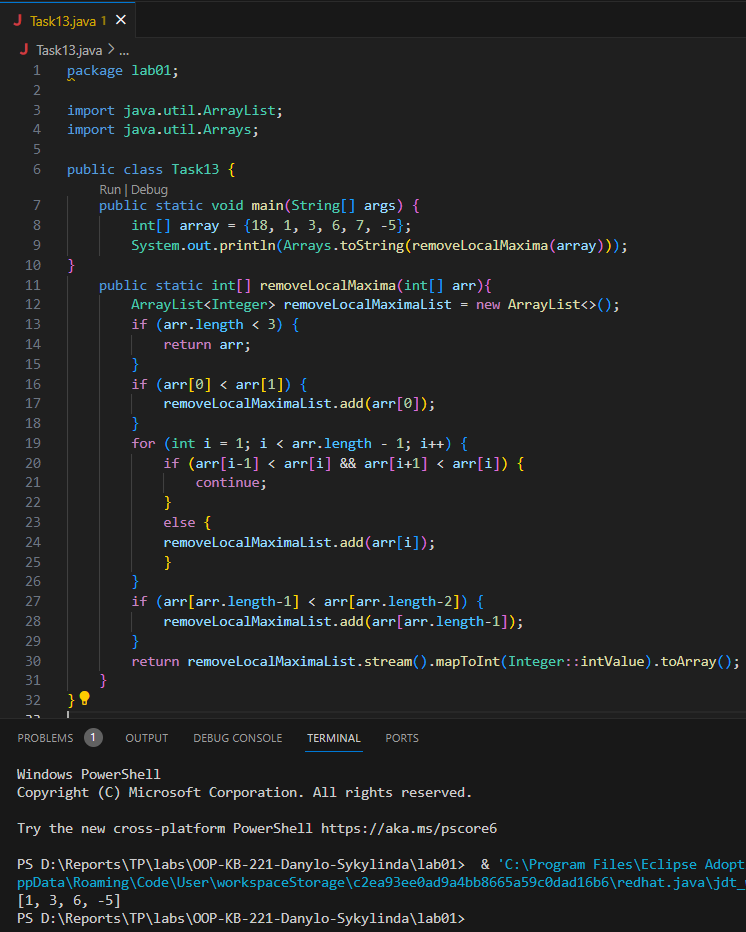


Рис. 20. Тринадцяте завдання. Нижче на фото перевірка роботи коду.

У main() я виводжу виклик методу removeLocalMaxima(), передавши в якості аргументу масив array, результат виклику перетворюю в строку. У методі removeLocalMaxima() реалізую створення масиву removeLocalMaximaList, у який будемо зберігати числа, які не є локальними максимумами.

Далі я реалізую перевірку, що якщо масив arr має менше трьох елементів, то просто повертаю масив arr. Далі прописую ще одну умову, якщо нульовий елемент масиву менше, ніж перший елемент масиву arr, то масиву removeLocalMaximaList додати нульовий елемент масиву arr, тому що якщо перший елемент масиву arr більше за нульовий, то перший елемент масиву arr є локальним максимумом, а за умовою задачі він має бути видаленим. Отже, додаємо нульовий елемент масиву arr, який треба нам залишити, до масиву removeLocalMaximaList.

Далі я створюю цикл for, який перебирає елементи масиву arr з індексу 1 до індексу, що дорівнює довжина масиву мінус один, у якому я додаю ще умову, що якщо елемент, що перед поточним елементом (arr[i-1]), що перебирається, є меншим за поточний елемент(arr[i]) і елемент, що є після поточного елементу(arr[i+1]) є меншим, ніж поточний елемент(arr[i]), то переходимо до наступної ітерації за допомогою continue. Інакше додаємо поточний елемент(arr[i]) до масиву removeLocalMaximaList. Ця перевірка перевіряє, чи є максимумом поточний елемент(arr[i]), якщо є, то переходить до наступної ітерації, інакше додає елемент до масиву removeLocalMaximaList. Ми пам’ятаємо чому так, оскільки ми до масиву removeLocalMaximaList записуємо вже значення, які не є локальними максимумами.

Потім робимо перевірку, що якщо останній елемент масиву arr([arr.length - 1]) менше, ніж передостанній елемент масиву arr([arr.length - 2]), то додати останній елемент масиву arr до масиву removeLocalMaximaList. Бо якщо останній елемент масиву arr менше передостаннього, то це означає, що останній елемент не є локальним максимумом, отже, ми його додаємо до масиву removeLocalMaximaList. Тепер необхідно повернути масив removeLocalMaximaList, але нам потрібно повернути як масив цілих чисел, а в нас removeLocalMaximaList є не масивом цілих чисел, а є масивом типу ArrayList<Integer>.

Тому за допомогою методу методу stream(), який є в Java перетворюємо масив removeLocalMaximaList у потік цілих чисел, далі до цього всього використовуємо функцію mapToInt(), у якому зіставляємо кожен елемент потоку(елемент є цілим числом Integer) з цілим числом(типу int), тобто, об’єкти Integer конвертуються до цілого числа типу int за допомогою методу mapToInt(). Потім все, що ми отримали після всіх маніпуляцій в потоці, зводимо до масиву за допомогою методу toArray(). І таким чином повернувши вираз, який я написав, отримуємо результат.

## Завдання 14

|  |  |
| --- | --- |
| 14 | Напишіть програму що містить два методи, що реалізують наступний функціонал:  void cycleSwap(int[] array) - Зсуває всі елементи в даному масиві вправо на 1 позицію. В цьому випадку останній елемент масиву стає першим. Наприклад, 1 3 2 7 4 стає 4 1 3 2 7.  void cycleSwap(int[] array, int shift) - Зсуває всі елементи в заданому масиві вправо на shift позицій. Гарантується, що значення зсуву невід'ємне і не більше за довжину масиву. Наприклад, 1 3 2 7 4 зі зсувом 3 стає 2 7 4 1 3. |

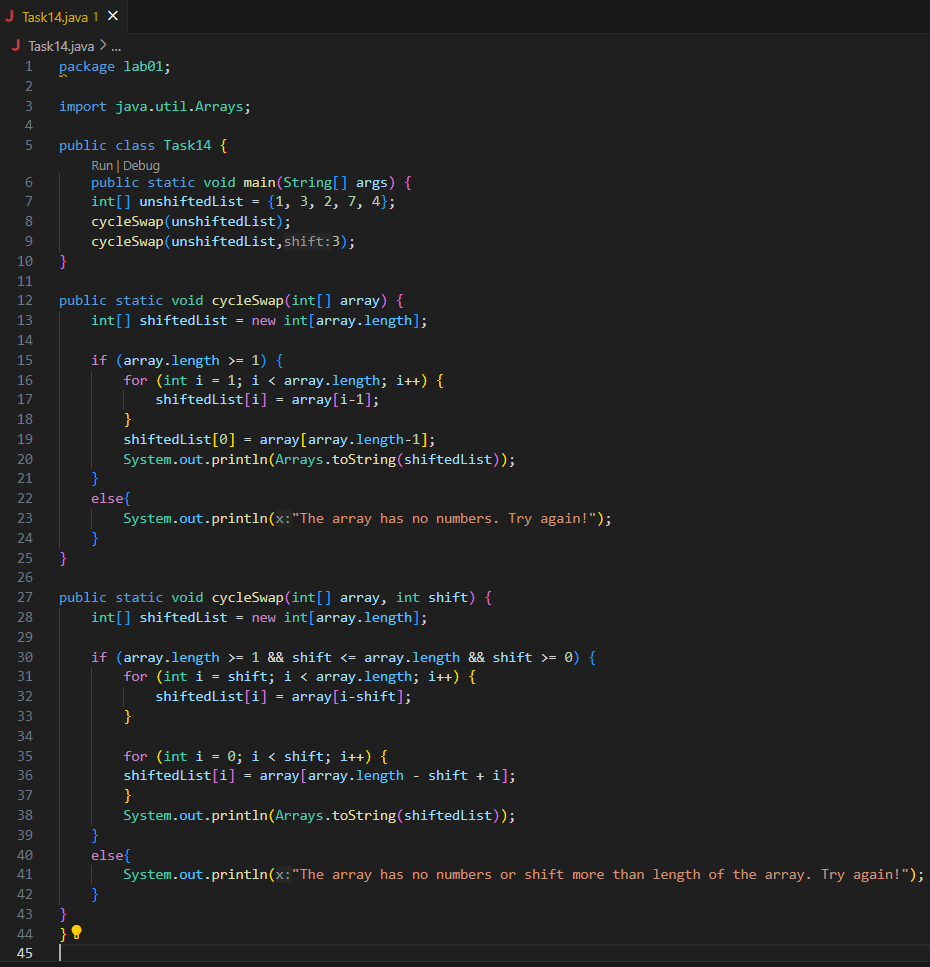


Рис. 21. Чотирнадцяте завдання.

У main() створено заданий масив та виклики методів cycleSwap(unshiftedList) та cycleSwap(unshiftedList, 3).

Реалізуємо метод cycleSwap(), який отримує як аргумент масив.

У цьому методі я створюю масив цілих чисел shiftedList довжиною як довжина масиву аргументу array. Потім я прописав умову, що якщо довжина масиву array є більше або дорівнює 1, то за допомогою циклу for з першого елементу(i=1) робимо таке: елементу з індексом «і» з масиву shiftedList призначити значення елементу з індексом «і – 1» з масиву array. Чому з першого елементу ми перебираємо циклом? Оскільки нульовим елементом масиву shiftedList має бути останній елемент масиву array, саме тому ми починаємо з першого елементу масиву shiftedList, щоб потім розібратися з нулевим. Коли цикл for відпрацював, то нульовому елементу масиву shiftedList призначаємо останній елемент(індексом визначаємо так: array.length – 1) масиву array. І виводимо результат, перетворивши масив у строку за допомогою Array.toString().

Тепер реалізуємо метод cycleSwap(), який в якості аргументу отримує масив цілих чисел та зсув, який необхідно зробити для елементів цього масиву.

За аналогією, як попередньому методі, створюємо масив shiftedList. Далі робимо перевірку, що якщо довжина масиву array є більше або дорівнює 1 і зсув менше або дорівнює довжині масиву array і зсув є більше або дорівнює нулю, то через цикл for перебираємо з i-того елементу(при тому, що i = shift) масив shiftedList і елементу з індексом «і» з масиву shiftedList призначаємо елемент з індексом «i - shift» з масиву array. Це ми зсунули елементи масиву array, які тепер в новому масиві shiftedList будуть в кінці.

Тепер потрібно ще один цикл for для того, щоб призначити у новому масиві shiftedList елементи, які мають бути спереду. У циклі for будемо призначати значення елементам масиву shiftedList з нульового індексу до індексу shift невключно, тобто, елементу з індексом «і» з масиву shiftedList призначити елемент з індексом «array.length – shift + i» з масиву array. Таким чином, елементи масиву array, що були в кінці масиву, то тепер будуть в новому масиві shiftedList на початку масиву. Виводимо результат, як при розробці попереднього методу.

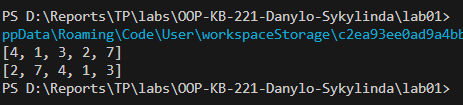


Рис. 22. Продовження чотирнадцятого завдання. Перевірка коду завдання.

# ВИСНОВОК

Я ознайомився з матеріалом, виконав всі завдання, що були подані до лабораторної роботи. Протестував коди всіх завдань і переконався, що вони задовольняють вимоги до завдань. Краще ознайомився з основами мови програмування Java, покращив свої навички в вирішенні задач. Завантажив всі виконані завдання та звіт лабораторної роботи на Github та Moodle.