# Лабораторна робота №1

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

**Мета роботи**: 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 лабороторної 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):

|  |  |
| --- | --- |
| 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. |
| 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 |
| 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. |
| 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. |
| 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 |
| 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 |
| 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 |
| 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 |
| 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 |
| 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. |
| 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. |
| 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] |
| 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] |
| 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. |
|  |  |
|  |  |

Сформуйте звіт який містить код програм та результати тестування.

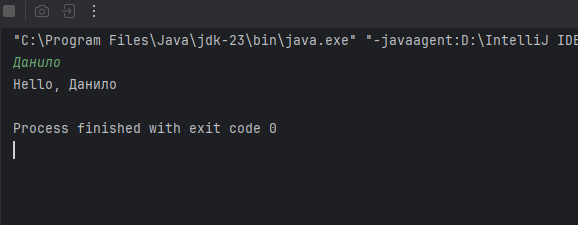
Надайте посилання на github для перевірки.

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.

Код:

package lab1;  
import java.util.Scanner;  
  
public class task1 {  
 public static void main(String[] args) {  
 // Створюємо об'єкт Scanner для зчитування введених даних  
 Scanner scanner = new Scanner(System.*in*);  
  
 // Зчитуємо рядок із консолі  
 String input = scanner.nextLine();  
  
 // Виводимо повідомлення  
 System.*out*.println("Hello, " + input);  
  
 // Закриваємо Scanner  
 scanner.close();  
 }  
}

Результат:



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

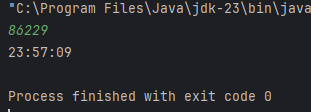
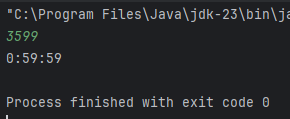
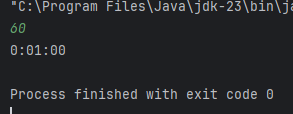
Введення даних: 86229

Результат виконання: 23:57:09

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task2 { public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 int secondsInput = scanner.nextInt();  
  
 int hours = (secondsInput / 3600) % 24;  
 int minutes = (secondsInput % 3600) / 60;  
 int seconds = secondsInput % 60;  
  
 System.*out*.printf("%d:%02d:%02d%n", hours, minutes, seconds);  
}  
}

Результат:



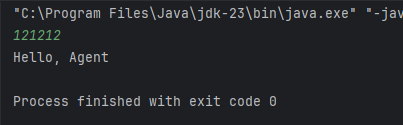
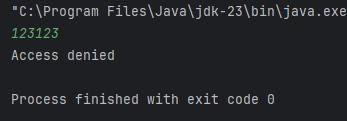
3) Напишіть програму, яка запитує на вхід число-пароль. Якщо воно дорівнює секретній кількості, програма виводить: "Hello, Agent", інакше друкує: "Access denied".

Секретний пароль зберігається в final static int password.

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task3 {final static int *password* = 121212; // секретний пароль  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 int input = scanner.nextInt();  
  
 if (input == *password*) {  
 System.*out*.println("Hello, Agent");  
 } else {  
 System.*out*.println("Access denied");  
 }  
 }  
}

Результат:

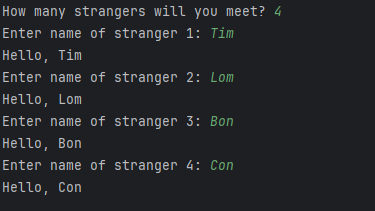
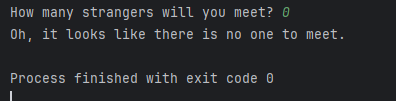
4) Напишіть просту програму, яка запитує кількість незнайомців, з якими потрібно зустрітися, рядково читає імена незнайомців, по рядках виводить: "Hello, (ім'я незнайомця)" для кожного незнайомця.

Доповнити програму перевірками на кількість, що є від’ємною, та рівна 0.

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task4 {public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 System.*out*.print("How many strangers will you meet? ");  
 int count = scanner.nextInt();  
 scanner.nextLine(); // зчитує залишок після числа  
  
 if (count < 0) {  
 System.*out*.println("Error: number of strangers cannot be negative.");  
 } else if (count == 0) {  
 System.*out*.println("Oh, it looks like there is no one to meet.");  
 } else {  
 for (int i = 0; i < count; i++) {  
 System.*out*.print("Enter name of stranger " + (i + 1) + ": ");  
 String name = scanner.nextLine();  
 System.*out*.println("Hello, " + name);  
 }  
 }  
}  
}

Результат:

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

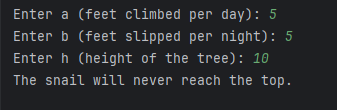
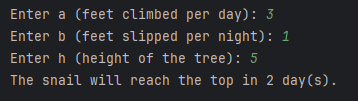
Введення даних: 4 4 10

Результат: Impossible

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task5 { public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 System.*out*.print("Enter a (feet climbed per day): ");  
 int a = scanner.nextInt();  
  
 System.*out*.print("Enter b (feet slipped per night): ");  
 int b = scanner.nextInt();  
  
 System.*out*.print("Enter h (height of the tree): ");  
 int h = scanner.nextInt();  
  
 if (a <= b && a < h) {  
 System.*out*.println("The snail will never reach the top.");  
 }  
 else {  
 int effectiveProgress = a - b;  
 int days = (h - a + effectiveProgress - 1) / effectiveProgress + 1;  
 System.*out*.println("The snail will reach the top in " + days + " day(s).");  
 }  
}  
}

Результат:

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**:

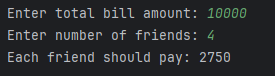
Введення даних: 1000 5

Результат: 220

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task6 {public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 System.*out*.print("Enter total bill amount: ");  
 double bill = scanner.nextDouble();  
  
 System.*out*.print("Enter number of friends: ");  
 int friends = scanner.nextInt();  
  
 if (bill < 0) {  
 System.*out*.println("Error: bill amount cannot be negative.");  
 } else if (friends <= 0) {  
 System.*out*.println("Error: number of friends must be greater than zero.");  
 } else {  
 double totalWithTip = bill \* 1.10; // додати 10% чайових  
 int share = (int) Math.*round*(totalWithTip / friends);  
 System.*out*.println("Each friend should pay: " + share);  
 }  
}  
}

Результат:



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.

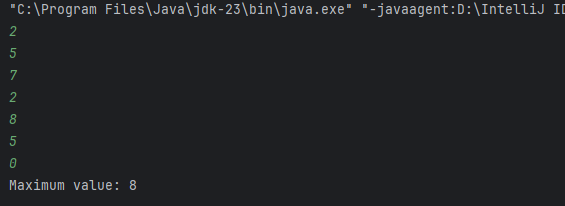
**Приклад**:

Введення даних: 2 4 6 9 2 4 5 0

Результат: 9

Код: package lab1;  
  
import java.util.Scanner;  
  
public class task7 {public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 int max = Integer.*MIN\_VALUE*;  
 while (true) {  
 int num = scanner.nextInt();  
 if (num == 0) {  
 break;  
 }  
 if (num > max) {  
 max = num;  
 }  
 }  
  
 System.*out*.println("Maximum value: " + max);  
}  
}

Результат:



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:**

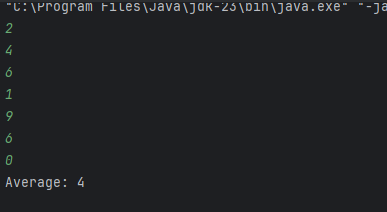
Введення даних: 1 2 3 4 5 0

Результат: 3

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task8 {public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 int sum = 0;  
 int count = 0;  
  
 while (true) {  
 int num = scanner.nextInt();  
 if (num == 0) {  
 break;  
 }  
 sum += num;  
 count++;  
 }  
  
 int average = sum / count;  
 System.*out*.println("Average: " + average);  
}  
}

Результат:



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

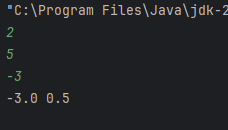
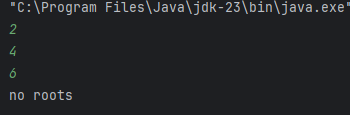
Введення даних: 2 2 2

Результат : no roots

Код:

package lab1;  
  
import java.util.Scanner;  
  
public class task9 {public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 double a = scanner.nextDouble();  
 double b = scanner.nextDouble();  
 double c = scanner.nextDouble();  
  
 double discriminant = b \* b - 4 \* a \* c;  
  
 if (discriminant < 0) {  
 System.*out*.println("no roots");  
 } else if (discriminant == 0) {  
 double x = -b / (2 \* a);  
 System.*out*.println(x);  
 } else {  
 double sqrtD = Math.*sqrt*(discriminant);  
 double x1 = (-b - sqrtD) / (2 \* a);  
 double x2 = (-b + sqrtD) / (2 \* a);  
 System.*out*.println(x1 + " " + x2);  
 }  
}  
}

Результат:

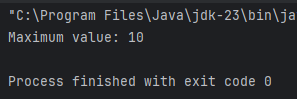


10) Напишіть програму, що містить метод max(…). Правильна реалізація має отримати масив значень типу int та повернути його максимальне значення. Вхідний масив гарантовано не буде порожнім або null. Метод max не повинен змінювати масив.

Код:

package lab1;  
  
public class task10 {  
  
 public static int max(int[] array) {  
 int maxValue = array[0];  
 for (int i = 1; i < array.length; i++) {  
 if (array[i] > maxValue) {  
 maxValue = array[i];  
 }  
 }  
 return maxValue;  
}  
  
 public static void main(String[] args) {  
 int[] sample = {3, 8, 2, 10, 5};  
 System.*out*.println("Maximum value: " + *max*(sample));  
 }  
}

Результат:

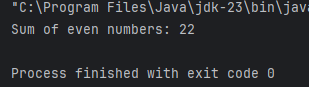


11) Напишіть програму, що містить метод sum. Правильна реалізація має отримати масив значень int та повернути суму парних чисел. Якщо цей масив дорівнює нулю або порожній, метод повертає 0. Метод sum не повинен змінювати масив.

Код:

package lab1;  
  
public class task11 {public static int sum(int[] array) {  
 if (array == null || array.length == 0) {  
 return 0;  
 }  
  
 int total = 0;  
 for (int num : array) {  
 if (num % 2 == 0) {  
 total += num;  
 }  
 }  
  
 return total;  
}  
  
 public static void main(String[] args) {  
 int[] data = {1, 4, 3, 6, 5, 12};  
 System.*out*.println("Sum of even numbers: " + *sum*(data));  
 }  
}

Результат:



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]

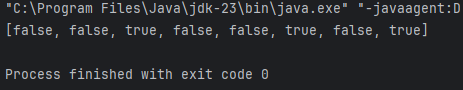
Результат: [false, false, true, false, false, true, false, true]

Код:

package lab1;  
  
import java.util.Arrays;  
  
public class task12 {public static boolean[] getSumCheckArray(int[] array) {  
 boolean[] result = new boolean[array.length];  
  
 // Перші два завжди false  
 result[0] = false;  
 result[1] = false;  
  
 for (int i = 2; i < array.length; i++) {  
 result[i] = array[i] == array[i - 1] + array[i - 2];  
 }  
  
 return result;  
}  
  
 public static void main(String[] args) {  
 int[] input = {1, -1, 0, 4, 6, 10, 15, 25};  
 boolean[] output = *getSumCheckArray*(input);  
 System.*out*.println(Arrays.*toString*(output));  
 }  
}

Результат:

{1, -1, 0, 4, 6, 10, 15, 25};



14)Напишіть програму що містить два методи, що реалізують наступний функціонал:

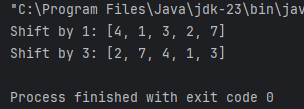
void cycleSwap(int[] array) - Зсуває всі елементи в даному масиві вправо на 1 позицію. В цьому випадку останній елемент масиву стає першим. Наприклад, 1 3 2 7 4 стає 4 1 3 2 7.

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.

Код:

package lab1;  
  
import java.util.Arrays;  
  
public class task14 {// Зсув на одну позицію  
 public static void cycleSwap(int[] array) {  
 if (array.length == 0) return;  
  
 int last = array[array.length - 1];  
 for (int i = array.length - 1; i > 0; i--) {  
 array[i] = array[i - 1];  
 }  
 array[0] = last;  
 }  
  
 // Зсув на shift позицій  
 public static void cycleSwap(int[] array, int shift) {  
 if (array.length == 0 || shift == 0) return;  
  
 shift = shift % array.length; // для безпеки  
  
 *reverse*(array, 0, array.length - 1);  
 *reverse*(array, 0, shift - 1);  
 *reverse*(array, shift, array.length - 1);  
 }  
  
 // Допоміжний метод для розвороту частини масиву  
 private static void reverse(int[] array, int start, int end) {  
 while (start < end) {  
 int temp = array[start];  
 array[start] = array[end];  
 array[end] = temp;  
 start++;  
 end--;  
 }  
 }  
  
 // Демонстрація  
 public static void main(String[] args) {  
 int[] data1 = {1, 3, 2, 7, 4};  
 *cycleSwap*(data1);  
 System.*out*.println("Shift by 1: " + Arrays.*toString*(data1)); // [4, 1, 3, 2, 7]  
  
 int[] data2 = {1, 3, 2, 7, 4};  
 *cycleSwap*(data2, 3);  
 System.*out*.println("Shift by 3: " + Arrays.*toString*(data2)); // [2, 7, 4, 1, 3]  
 }  
}

Результат:



https://github.com/yuniqwll/OOP