**Звіт до Лабораторної роботи №1**

**Тема:** Типи даних, умови, цикли та масиви.

**Мета роботи:** Використовуючи теоретичне підґрунтя про складні структури даних Списки, Словники Кортежі, та використовуючи існуючий код, доробити програму додавши функціонал, що буде вказано в завданні до лабораторної роботи.

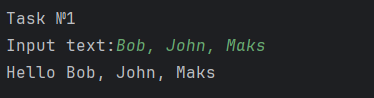
**Хід роботи**

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 програми:

import java.util.Scanner;  
  
public class task\_01 {  
 public static void main(String[] args) {  
 System.*out*.printf("Task №1\n");  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.printf("Input text:");  
 String new\_str = scan.nextLine();  
 System.*out*.println("Hello " + new\_str);  
 }  
}

Результат роботи програми:

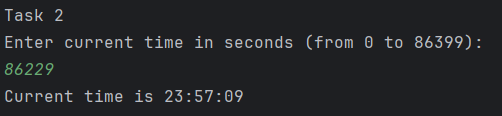


2. Напишіть програму, яка отримує на вхід ціле число. Це число є поточний добовий час, який обчислюється як кількість секунд, що пройшли з півночі. Потім програма повинна надрукувати значення екрану електронного годинника для цього часу. Вхідне значення отримується через System.in. Вихідне значення має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

Код програми:

import java.util.Scanner;  
  
public class task\_02 {  
 public static void main(String[] argv) {  
 System.*out*.println("Task 2");  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("Enter current time in seconds (from 0 to 86399):");  
 Integer input\_num = scan.nextInt();  
 if ((input\_num < 0)||(input\_num > 86399)){  
 System.*out*.println("Wrong input: " + input\_num + " time conversion is not possible");  
 return;  
 }  
  
 int hours = input\_num / 3600;  
 int times = input\_num - (hours \* 3600);  
 int minutes = times / 60;  
 int seconds = times - (minutes \* 60);  
  
 String str\_hours = "";  
 String str\_min = "";  
 String str\_seconds = "";  
 if (hours < 10){  
 str\_hours = "0" + hours;  
 }  
 else {  
 str\_hours = "" + hours;  
 }  
 if (minutes < 10){  
 str\_min = "0" + minutes;  
 }  
 else {  
 str\_min = "" + minutes;  
 }  
 if (seconds < 10){  
 str\_seconds = "0" + seconds;  
 }  
 else {  
 str\_seconds = "" + seconds;  
 }  
 System.*out*.println("Current time is "+str\_hours+":"+str\_min+":"+str\_seconds);  
 }  
}

Результат роботи програми:

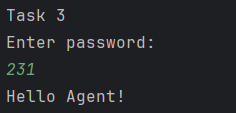


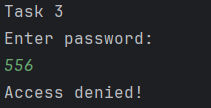
3. Напишіть програму, яка запитує на вхід число-пароль. Якщо воно дорівнює секретній кількості, програма виводить: "Hello, Agent", інакше друкує: "Access denied". Секретний пароль зберігається в final static int password.

Код програми:

import java.util.Scanner;  
  
public class task\_03 {  
 final static int *password* = 231;  
 public static void main(String[] argv){  
 System.*out*.println("Task 3");  
  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("Enter password:");  
 Integer int\_passwd = scan.nextInt();  
 if (int\_passwd == *password*){  
 System.*out*.println("Hello Agent!");  
 }  
 else {  
 System.*out*.println("Access denied!");  
 }  
 }  
}

Результат роботи програми:



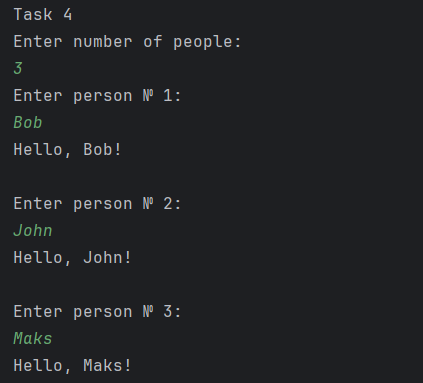


4. Напишіть просту програму, яка запитує кількість незнайомців, з якими потрібно зустрітися, рядково читає імена незнайомців, по рядках виводить: "Hello, (ім'я незнайомця)" для кожного незнайомця. Доповнити програму перевірками на кількість, що є від’ємною, та рівна 0.

Код програми:

import java.util.Scanner;  
  
public class task\_04 {  
 public static void main(String[] argv){  
 System.*out*.println("Task 4");  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("Enter number of people:");  
 int number = scan.nextInt();  
 String human = "";  
 if (number <= 0){  
 System.*out*.println("Wrong input");  
 return;  
 }  
  
 for (int i=1; i <= number; i++){  
 System.*out*.println("Enter person № " + i + ":");  
 Scanner scan2 = new Scanner(System.*in*);  
 human = scan2.nextLine();  
 System.*out*.println("Hello, " + human + "!\n");  
 }  
 }  
}

Результат роботи програми:

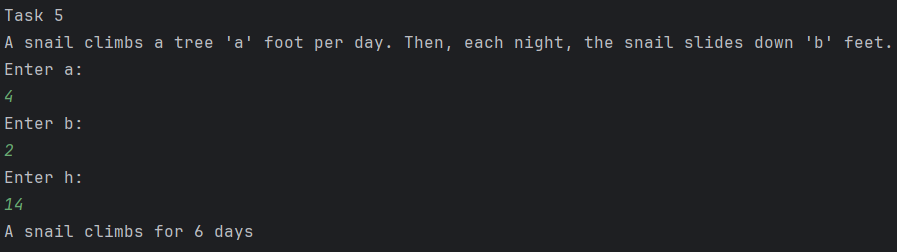


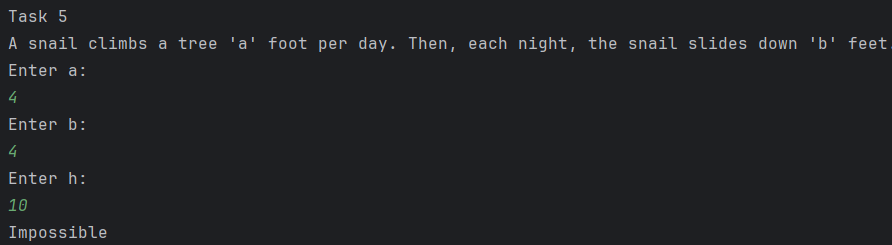
5. Наведено таку умову: равлик піднімається по дереву на a футів на день. Потім щоночі равлик сповзає вниз на b футів. Висота дерева — h футів. Напишіть програму з підрахунком кількості днів, які знадобляться равлику, щоб дістатися до вершини дерева. Програма читає a, b, h рядково. Вхідні значення гарантовано є додатніми цілими числами. Якщо равлик не може дістатися до вершини дерева, має виводитися повідомлення: Impossible

Код програми:

import java.util.Scanner;  
  
public class task\_05 {  
 public static void main(String[] argv) {  
 System.*out*.println("Task 5");  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("A snail climbs a tree 'a' foot per day. " +  
 "Then, each night, the snail slides down 'b' feet. " +  
 "The tree is 'h' feet tall.");  
 System.*out*.println("Enter a:");  
 int a = scan.nextInt();  
 System.*out*.println("Enter b:");  
 int b = scan.nextInt();  
 System.*out*.println("Enter h:");  
 int h = scan.nextInt();  
  
 if (a >= h){  
 System.*out*.println("A snail climbs for 1 day");  
 return;  
 }  
 if ((a <= b)&&(a < h)){  
 System.*out*.println("Impossible");  
 return;  
 }  
 if ((a > b)&&(a < h)){  
 int lift=0, days=0;  
 while (lift < h){  
 lift = lift + a;  
 days = days +1;  
 if (lift >= h){  
 System.*out*.println("A snail climbs for " + days + " days");  
 break;  
 }  
 lift = lift - b;  
 }  
 }  
 }  
}

Результат роботи програми:



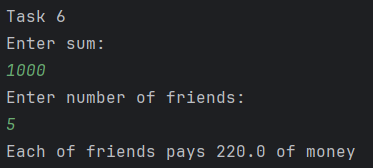


6. Дано така умова: компанія друзів відвідує ресторан. Вони вирішили розділити рахунок порівну та додати 10 відсотків від загальної суми рахунку як чайові. Далі друзі покривають рівними частинами загальну суму платежу. Напишіть програму, яка зчитує загальну суму рахунку та кількість друзів, а потім виводить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

Код програми:

import java.util.Scanner;  
  
public class task\_06 {  
 public static void main(String[] args) {  
 System.*out*.println("Task 6");  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("Enter sum:");  
 double sum = scan.nextDouble();  
 if (sum <= 0){  
 System.*out*.println("Wrong input");  
 return;  
 }  
 System.*out*.println("Enter number of friends:");  
 double friends = scan.nextDouble();  
 if (friends <= 0){  
 System.*out*.println("Wrong input");  
 return;  
 }  
 double each\_sum = (sum \* 1.1) / friends;  
 System.*out*.println("Each of friends pays " + each\_sum + " of money");  
 }  
}

Результат роботи програми:

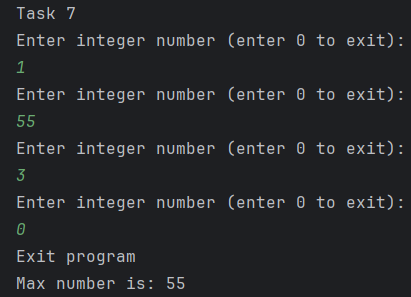


7. Напишіть програму, яка зчитує послідовність цілих значень зі стандартного введення і знаходить максимальне з них. Ви повинні зчитувати значення послідовності, доки наступне не стане 0. Нульове значення означає кінець вхідної послідовності та не є її елементом. Послідовність гарантовано містить хоча б одне значення.

Код програми:

import java.util.Scanner;  
  
public class task\_07 {  
 public static void main(String[] args) {  
 System.*out*.println("Task 7");  
 int num1 = 0;  
 Scanner scan = new Scanner(System.*in*);  
 while (true){  
 System.*out*.println("Enter integer number (enter 0 to exit):");  
 int num = scan.nextInt();  
 if (num == 0){  
 System.*out*.println("Exit program");  
 break;  
 }  
 if (num > num1){  
 num1 = num;  
 }  
 }  
 System.*out*.println("Max number is: " + num1);  
 }  
}

Результат роботи програми:

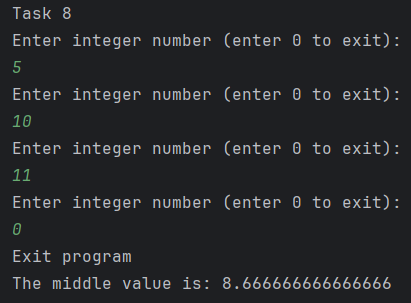


8. Напишіть програму, яка зчитує послідовність цілих значень зі стандартного введення та знаходить середнє значення. Ви повинні зчитувати значення послідовності, доки наступне не стане 0. Нульове значення означає кінець вхідної послідовності та не є її частиною. Послідовність гарантовано містить хоча б одне значення.

Код програми:

import java.util.Scanner;  
  
public class task\_08 {  
 public static void main(String[] args) {  
 System.*out*.println("Task 8");  
 Scanner scan = new Scanner(System.*in*);  
 double num1 = 0, i = 0;  
 while (true){  
 System.*out*.println("Enter integer number (enter 0 to exit):");  
 double num = scan.nextDouble();  
 if (num == 0){  
 System.*out*.println("Exit program");  
 break;  
 }  
 num1 = num1 + num;  
 i = i + 1;  
 }  
 double middle = num1 / i;  
 System.*out*.println("The middle value is: " + middle);  
 }  
}

Результат роботи програми:

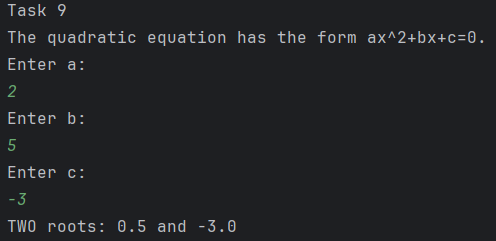


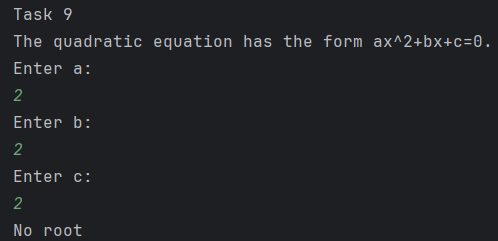
9. Напишіть програму для вирішення квадратних рівнянь. При виконанні врахуйте залежність значення дискримінанту до кількості результатів

Код програми:

import java.util.Scanner;  
import static java.lang.Math.*sqrt*;  
  
public class task\_09 {  
 public static void main(String[] argv){  
 System.*out*.println("Task 9");  
 Scanner scan = new Scanner(System.*in*);  
 System.*out*.println("The quadratic equation has the form ax^2+bx+c=0.");  
 System.*out*.println("Enter a:");  
 float a = scan.nextFloat();  
 System.*out*.println("Enter b:");  
 float b = scan.nextFloat();  
 System.*out*.println("Enter c:");  
 float c = scan.nextFloat();  
  
 float d = b\*b - 4\*a\*c;  
 if (d < 0){  
 System.*out*.println("No root");  
 }  
 else if (d == 0){  
 float x1 = (-b) / 2\*a;  
 System.*out*.println("ONE root: " + x1);  
 }  
 else if (d > 0){  
 float x1 = ((-b) + (float)*sqrt*(d)) / (2\*a);  
 float x2 = ((-b) - (float)*sqrt*(d)) / (2\*a);  
 System.*out*.println("TWO roots: " + x1 + " and " + x2);  
 }  
 }  
}

Результат роботи програми:



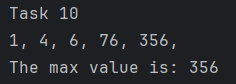


10. Напишіть програму, що містить метод max(…). Правильна реалізація має отримати масив значень типу int та повернути його максимальне значення. Вхідний масив гарантовано не буде порожнім або null. Метод max не повинен змінювати масив.

Код програми:

public class task\_10 {  
 public static void main(String[] argv){  
 System.*out*.println("Task 10");  
 int [] numbers = {1,4,6,76,356};  
 for (int i = 0; i < numbers.length; i++){  
 System.*out*.print(numbers[i] + ", ");  
 }  
  
 System.*out*.println("\nThe max value is: " + *max*(numbers));  
 }  
  
 public static int max(int[] array){  
 int max\_value = 0;  
 for (int i = 0; i < array.length; i++){  
 if (array[i] > max\_value){  
 max\_value = array[i];  
 }  
 }  
 return max\_value;  
 }  
}

Результат роботи програми:

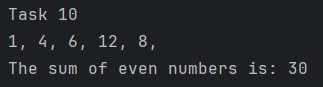


11. Напишіть програму, що містить метод sum. Правильна реалізація має отримати масив значень int та повернути суму парних чисел. Якщо цей масив дорівнює нулю або порожній, метод повертає 0. Метод sum не повинен змінювати масив.

Код програми:

public class task\_11 {  
 public static void main(String[] argv){  
 System.*out*.println("Task 10");  
 int [] numbers = {1,4,6,12,8};  
 for (int i = 0; i < numbers.length; i++){  
 System.*out*.print(numbers[i] + ", ");  
 }  
  
 System.*out*.println("\nThe sum of even numbers is: " + *sum*(numbers));  
 }  
  
 public static int sum(int[] array){  
 int sum1 = 0;  
 for (int i = 0; i < array.length; i++){  
 if (array[i] % 2 == 0){  
 sum1 = sum1 + array[i];  
 }  
 }  
 return sum1;  
 }  
}

Результат роботи програми:

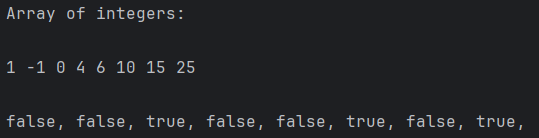


12. Напишіть програму, що містить метод getSumCheckArray. Правильна реалізація повинна отримати масив значень int і повернути масив логічних значень, де кожен елемент є результатом перевірки: чи є вихідний елемент сумою двох попередніх елементів у даному масиві. Гарантується, що довжина масиву становить 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.

Код програми:

public class task\_12 {  
 static boolean[] *bool\_array* = {false, false, false, false, false, false, false, false};  
 public static void main(String[] argv){  
 int[] numbers = {1, -1, 0, 4, 6, 10, 15, 25};  
 System.*out*.println("Array of integers:\n");  
 for (int i=0; i < numbers.length; i++){  
 System.*out*.print(numbers[i] + " ");  
 }  
 System.*out*.println("\n");  
 *getSumCheckArray*(numbers);  
 for (int i=0; i < *bool\_array*.length; i++){  
 System.*out*.print(*bool\_array*[i] + ", ");  
 }  
 }  
  
 public static void getSumCheckArray(int[] array){  
 for (int i=2; i < array.length; i++){  
 if (array[i] == array[i-1] + array[i-2]){  
 *bool\_array*[i] = true;  
 }  
 }  
 }  
}

Результат роботи програми:



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.\*.

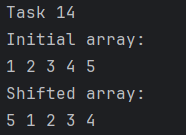
14. Напишіть програму що містить два методи, що реалізують наступний функціонал:

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.

Код програми:

public class task\_14\_1 {  
 static int[] *numbers* = {1, 2, 3, 4, 5};  
 static int[] *sortNumbers* = {0, 0, 0, 0, 0};  
  
 public static void main(String[] args){  
 System.*out*.println("Task 14");  
 System.*out*.println("Initial array:");  
 for (int i=0; i < *numbers*.length; i++){  
 System.*out*.print(*numbers*[i] + " ");  
 }  
 System.*out*.println("\nShifted array:");  
 *cycleSwap*(*numbers*);  
 }  
  
 public static void cycleSwap(int[] array){  
 for (int i=0; i < array.length; i++){  
 if (i == array.length - 1){  
 *sortNumbers*[0] = array[i];  
 break;  
 }  
 *sortNumbers*[i+1] = *numbers*[i];  
 }  
 for (int i=0; i < *sortNumbers*.length; i++){  
 System.*out*.print(*sortNumbers*[i] + " ");  
 }  
 }  
}

Результат роботи програми:

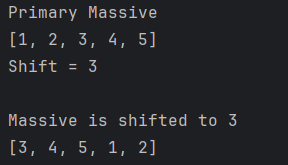


void cycleSwap(int[] array, int shift) - Зсуває всі елементи в заданому масиві вправо на shift позицій. Гарантується, що значення зсуву невід'ємне і не більше за довжину масиву. Наприклад, 1 3 2 7 4 зі зсувом 3 стає 2 7 4 1 3.

Код програми:

import java.util.Arrays;  
  
public class task\_14\_2 {  
 public static void main(String[] args) {  
 int[] array = {1, 2, 3, 4, 5};  
 System.*out*.println("Primary Massive");  
 System.*out*.println(Arrays.*toString*(array));  
 int shift = 3;  
 System.*out*.println("Shift = " + shift);  
 *cycleSwap*(array, shift);  
 }  
  
 public static void cycleSwap(int[] array, int shift) {  
 int len = array.length-1;  
 int[] ar1 = {0, 0, 0, 0, 0};  
 if (len == 0 || shift == 0 || shift == len+1) {  
 System.*out*.println("Massive is NOT Changed");  
 return;  
 }  
 int curElem = 0;  
 for(int i=0; i <= len; i++){  
 if ((i + shift) <= len){  
 curElem = array[i];  
 ar1[i+shift]=curElem;  
 }  
 int rest=0;  
 if ((i + shift) > len){  
 rest=i + shift - len;  
 curElem=array[i];  
 ar1[rest -1]=curElem;  
 }  
 }  
 System.*out*.println("\nMassive is shifted to " + shift);  
 System.*out*.println(Arrays.*toString*(ar1));  
 }  
}

Результат роботи програми:



**Висновки.** Під час виконання цієї лабораторної роботи я вивчив: що таке типи даних, що використовуються у мові Java, правила використання умовних переходів, написання циклів та використання масивів.

Посилання на GitHub репозиторій:

<https://github.com/diachenkom/OOP-KB-231-Diachenko-Maksym>