**Class, Objects**

1.  Create Account A and Account B with an initial balance of 5000 and 2500 respectively. Transfer amount of 1500 from Account A to B and an amount of 3000 from Account B to A.

Print the receipt with the following details after each transaction

**Output:**

Account id: 12344,

Name: XXXX,

Account Balance: Rs.\_\_\_\_

**CODE:**

import java.util.Scanner;

class BankAccount {

private int accountId;

private String name;

private double balance;

public BankAccount(int accountId, String name, double balance) {

this.accountId = accountId;

this.name = name;

this.balance = balance;

}

public double getBalance() {

return balance;

}

public void transferTo(BankAccount recipient, double amount) {

if (balance >= amount) {

balance -= amount;

recipient.balance += amount;

printReceipt(recipient, amount, "Transfer to");

} else {

System.out.println("Insufficient funds for the transfer.");

}

}

private void printReceipt(BankAccount recipient, double amount, String transactionType) {

System.out.println("Account id: " + accountId);

System.out.println("Name: " + name);

System.out.println("Transaction Type: " + transactionType);

System.out.println("Amount: Rs. " + amount);

System.out.println("Recipient Account id: " + recipient.accountId);

System.out.println("Recipient Name: " + recipient.name);

System.out.println("Updated Account Balance: Rs. " + balance);

System.out.println("------------------------------");

}

}

public class Transaction {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter details for Account A:");

System.out.print("Account id: ");

int accountIdA = scanner.nextInt();

System.out.print("Name: ");

String nameA = scanner.next();

System.out.print("Initial Balance: ");

double initialBalanceA = scanner.nextDouble();

System.out.println("\nEnter details for Account B:");

System.out.print("Account id: ");

int accountIdB = scanner.nextInt();

System.out.print("Name: ");

String nameB = scanner.next();

System.out.print("Initial Balance: ");

double initialBalanceB = scanner.nextDouble();

BankAccount accountA = new BankAccount(accountIdA, nameA, initialBalanceA);

BankAccount accountB = new BankAccount(accountIdB, nameB, initialBalanceB);

System.out.println("\nEnter transfer amounts:");

System.out.print("Transfer amount from A to B: ");

double transferAmountAB = scanner.nextDouble();

accountA.transferTo(accountB, transferAmountAB);

System.out.print("Transfer amount from B to A: ");

double transferAmountBA = scanner.nextDouble();

accountB.transferTo(accountA, transferAmountBA);

scanner.close();

}

}

2.  Given an array and a partition size, you have to partition the array with that value , then we will specify the partition order, you have to merge based on that order

**Input:**

Array : 1 2 3 4 5

Partition size 2 (so the array will be partitioned as **1 2**, 3 4,**5**)

Partition order 3 2 1

**Output:**

5 3 4 1 2

**CODE:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

public class Array {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the array elements (comma-separated): ");

String[] arrayInput = scanner.nextLine().split(",");

int[] array = new int[arrayInput.length];

for (int i = 0; i < arrayInput.length; i++) {

array[i] = Integer.parseInt(arrayInput[i].trim());

}

System.out.print("Enter the partition size: ");

int partitionSize = scanner.nextInt();

System.out.print("Enter the partition order (comma-separated): ");

String[] orderInput = scanner.next().split(",");

int[] partitionOrder = new int[orderInput.length];

for (int i = 0; i < orderInput.length; i++) {

partitionOrder[i] = Integer.parseInt(orderInput[i].trim());

}

List<List<Integer>> partitions = partitionArray(array, partitionSize);

List<Integer> result = mergePartitions(partitions, partitionOrder);

System.out.println("Merged and ordered partitions:");

for (int num : result) {

System.out.print(num + " ");

}

}

private static List<List<Integer>> partitionArray(int[] array, int partitionSize) {

List<List<Integer>> partitions = new ArrayList<>();

for (int i = 0; i < array.length; i += partitionSize) {

List<Integer> partition = new ArrayList<>();

for (int j = i; j < Math.min(i + partitionSize, array.length); j++) {

partition.add(array[j]);

}

partitions.add(partition);

}

return partitions;

}

private static List<Integer> mergePartitions(List<List<Integer>> partitions, int[] partitionOrder) {

List<Integer> result = new ArrayList<>();

for (int order : partitionOrder) {

List<Integer> partition = partitions.get(order - 1);

result.addAll(partition);

}

return result;

}

}

3. A palindrome number - number that remains the same after reversing each digit of that number. A prime number - number that is divisible by only one or itself. A number that satisfies both the properties is said to be PalPrime Number.

Create a class PalPrime with a parameterised constructor PalPrime(int number, String message).

Given an positive integer array of numbers, you have to traverse the array and print the message “Number \_\_\_ is Prime/Palindrome/PalPrime”.

Note: Message should be printed via constructor of PalPrime class.

**Input** :

Array: [1, 34543, 565, 727, 10099]

**Output** -> Predict the output

**CODE:**

import java.util.Scanner;

class PalPrime {

private int number;

private String message;

public PalPrime(int number, String message) {

this.number = number;

this.message = message;

printMessage();

}

private void printMessage() {

System.out.println("Number " + number + " is " + message);

}

public static boolean isPalindrome(int num) {

int originalNum = num;

int reversedNum = 0;

while (num != 0) {

int digit = num % 10;

reversedNum = reversedNum \* 10 + digit;

num /= 10;

}

return originalNum == reversedNum;

}

public static boolean isPrime(int num) {

if (num <= 1) {

return false;

}

for (int i = 2; i <= Math.sqrt(num); i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

}

public class Palindrome {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements in the array: ");

int n = scanner.nextInt();

int[] numbers = new int[n];

System.out.println("Enter the elements of the array:");

for (int i = 0; i < n; i++) {

numbers[i] = scanner.nextInt();

}

for (int number : numbers) {

boolean isPalPrime = PalPrime.isPalindrome(number) && PalPrime.isPrime(number);

if (isPalPrime) {

new PalPrime(number, "PalPrime");

} else if (PalPrime.isPalindrome(number)) {

new PalPrime(number, "Palindrome");

} else if (PalPrime.isPrime(number)) {

new PalPrime(number, "Prime");

} else {

new PalPrime(number, "Neither Palindrome nor Prime");

}

}

scanner.close();

}

}