**Java Array: Exercises, Practice, Solution**

\* Write a Java program to sort a numeric array and a string array.

Arrays.sort(my\_array1);

System.out.println("Sorted numeric array : "+Arrays.toString(my\_array1));

Arrays.sort(my\_array2);

System.out.println("Sorted string array : "+Arrays.toString(my\_array2));

\* Write a Java program to sort a given array in reverse order.

Integer array1[] = new Integer[]{1, 4, 17, 7, 25, 3, 100};

Arrays.sort(array1, Collections.reverseOrder());

\* Write a Java program to sum values of an array.

int my\_array[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

int sum = 0;

for (int i : my\_array)

sum += i;

System.out.println("The sum is " + sum);

\* Write a Java program to calculate the average value of array elements.

int[] numbers = new int[]{20, 30, 25, 35, -16, 60, -100};

int sum = 0;

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

sum = sum + numbers[i];

double average = sum / numbers.length;

\* Write a Java program to test if an array contains a specific value.

I can check it in two ways...

**1)**

int myArray[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

ArrayList<int> list = Arrays.asList(myArray);

boolean b = list.contains(6);

or boolean b = (Arrays.asList(myArray)).contains(6);

: **int** myArray[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

List list = Arrays.*asList*(my\_array);

ArrayList list1 = (ArrayList) Arrays.*asList*(myArray);

**2)**

public static boolean contains(int[] arr, int item) {

for (int n : arr) {

if (item == n) {

return true; //after returning a true/false code stop execution

}

}

return false;

}

public static void main(String[] args) {

int[] my\_array1 = {1789, 2035, 1899, 1456, 2013};

System.out.println(contains(my\_array1, 2013));

System.out.println(contains(my\_array1, 2015));

}

Sample Output:

true

false

\* Write a Java program to remove a specific element from an array.

int[] my\_array = {25, 14, 56, 15, 36, 56, 77, 18, 29, 49};

int removeIndex = 1;

for(int i = removeIndex; i < my\_array.length -1; i++){

my\_array[i] = my\_array[i + 1];

}

// We cannot alter the size of an array , after the removal, the last and second last element in the array will exist twice

\* Write a Java program to copy an array by iterating the array.

int[] my\_array = {25, 14, 56, 15, 36, 56, 77, 18, 29, 49};

int[] new\_array = new int[10];

for(int i=0; i < my\_array.length; i++) {

new\_array[i] = my\_array[i];

}

\* Write a Java program to insert an element (specific position) into an array.

int[] my\_array = {25, 14, 56, 15, 36, 56, 77, 18, 29, 49};

int Index\_position = 2;

int newValue = 5;

for(int i=my\_array.length-1; i > Index\_position; i--){

my\_array[i] = my\_array[i-1];

}

my\_array[Index\_position] = newValue;

}

Sample Output:

Original Array : [25, 14, 56, 15, 36, 56, 77, 18, 29, 49]

New Array: [25, 14, 5, 56, 15, 36, 56, 77, 18, 29]

\* Write a Java program to reverse an array of integer values.

int[] my\_array1 = {1789, 2035, 1899, 1456, 2013, 1458};

for(int i = 0; i < my\_array1.length / 2; i++)

{

int temp = my\_array1[i];

my\_array1[i] = my\_array1[my\_array1.length - i - 1];

my\_array1[my\_array1.length - i - 1] = temp;

}

System.out.println("Reverse array : "+Arrays.toString(my\_array1));

}

OR

Integer [] myArray = {1,3,5,7,9};

System.out.println("Original Array:" + Arrays.asList(myArray));

Collections.reverse(Arrays.asList(myArray));

System.out.println("Reversed Array:" + Arrays.asList(myArray));

\* Write a Java program to find duplicate values in an array of integer values.

int[] my\_array = {1, 2, 5, 5, 6, 6, 7, 2};

for (int i = 0; i < my\_array.length-1; i++)

{

for (int j = i+1; j < my\_array.length; j++)

{

if ((my\_array[i] == my\_array[j]) && (i != j))

{

System.out.println("Duplicate Element : "+my\_array[j]);

}

}

}

\* Write a Java program to find duplicate values in an array of string values.

String[] my\_array = {"bcd", "abd", "jude", "bcd", "oiu", "gzw", "oiu"};

for (int i = 0; i < my\_array.length-1; i++){

for (int j = i+1; j < my\_array.length; j++){

if( (my\_array[i].equals(my\_array[j])) && (i != j) )

{

System.out.println("Duplicate Element is : "+my\_array[j]);

}

}

}

\* Write a Java program to find common elements between two arrays (string values).

String[] array1 = {"Python", "JAVA", "PHP", "C#", "C++", "SQL"};

String[] array2 = {"MySQL", "SQL", "SQLite", "Oracle", "PostgreSQL", "DB2", "JAVA"};

HashSet<String> set = new HashSet<String>();

for (int i = 0; i < array1.length; i++)

{

for (int j = 0; j < array2.length; j++)

{

if(array1[i].equals(array2[j]))

{

set.add(array1[i]);

}

}

}

System.out.println("Common element : "+(set));

\* Write a Java program to convert an array to an ArrayList.

String[] my\_array = new String[] {"Python", "JAVA", "PHP", "Perl", "C#", "C++"};

ArrayList<String> list = new ArrayList<String>(Arrays.asList(my\_array));  
OR

: **int** myArray[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

List list = Arrays.*asList*(my\_array);

ArrayList list1 = (ArrayList) Arrays.*asList*(myArray); // type casting

ArrayList <Integer> list2 = (ArrayList) Arrays.*asList*(myArray);

\* Write a Java program to find common elements in three sorted arrays.

ArrayList<Integer> common = new ArrayList<Integer>();

int array1[] = {2, 4, 8};

int array2[] = {2, 3, 4, 8, 10, 16};

int array3[] = {4, 8, 14, 40};

int x = 0, y = 0, z = 0;

while (x < array1.length && y < array2.length && z < array3.length){

if (array1[x] == array2[y] && array2[y] == array3[z]){

common.add(array1[x]);

x++;

y++;

z++;

}

else if (array1[x] < array2[y])

x++;

else if (array2[y] < array3[z])

y++;

else

z++;

}

\* Write a Java program to get the difference between the largest and smallest values in an array of integers.

int[] array\_nums = {5, 7, 2, 4, 9};

int max\_val = array\_nums[0];

int min = array\_nums[0];

for(int i = 1; i < array\_nums.length; i++)

{

if(array\_nums[i] > max\_val)

max\_val = array\_nums[i];

else if(array\_nums[i] < min)

min = array\_nums[i];

}

\* Write a Java program to concatenate a given string to the end of another string.

String str1 = "PHP Exercises and ";

String str2 = "Python Exercises";

String str3 = str1.concat(str2);

Output : PHP Exercises and Python Exercises

\* Write a Java program to create a String object with a character array.

char[] c = new char[] {'1', '5', '7', '2'};

String str = String.copyValueOf(c, 1, 3); **//c, 1, 3 means 1 to 3 index**

System.out.println("\nThe book contains " + str +" pages.\n");

Output : The book contains 572 pages.

**Java Collection: ArrayList Exercises**

\* Write a Java program to create a new array list, add some elements (string) and print out the collection.

List<String> list\_Strings = new ArrayList<String>();

list\_Strings.add("Red");

list\_Strings.add("Green");

list\_Strings.add("Orange");

list\_Strings.add("White");

list\_Strings.add("Black");

System.out.println(list\_Strings);

// Print the list

for (String element : list\_Strings) {

System.out.println(element);

}

\* Write a Java program to insert an element into the array list at the first position.

// Now insert a color at the first and last position of the list

list\_Strings.add(0, "Pink");

list\_Strings.add(5, "Yellow");

\* Write a Java program to retrieve an element (at a specified index) from a given array list.

// Retrive the first and third element

String element = list\_Strings.get(0);

element = list\_Strings.get(2);

\* Write a Java program to update specific array element by given element.

// Update the third element with "Yellow"

list\_Strings.set(2, "Yellow");

\* Write a Java program to remove the third element from an array list.

// Remove the third element from the list.

list\_Strings.remove(2);

\* Write a Java program to search an element in a array list.

// Search the value Red

if (list\_Strings.contains("Red")) {

System.out.println("Found the element");

} else {

System.out.println("There is no such element");

}

\* Write a Java program to sort a given array list.

Collections.sort(list\_Strings);

\* Write a Java program to copy one array list into another.

List<String> List1 = new ArrayList<String>();

List1.add("1");

List1.add("2");

List<String> List2 = new ArrayList<String>();

List2.add("A");

List2.add("B");

// Copy List2 to List1. But list has to be same size.

Collections.copy(List1, List2);

System.out.println("Copy List to List2,\nAfter copy:");

System.out.println("List1: " + List1);

System.out.println("List2: " + List2);

// Copy List2 to sublist with a smart idea.

List<String> sublist = List2.subList(0, List2.size());

OR

List<String> sublist = **new** ArrayList<String>(List1);

OR

List<String> sublist = **new** ArrayList<String>();

sublist.addAll(List1);

\* Write a Java program to reverse elements in a array list.

Collections.reverse(list\_Strings);

\* Write a Java program to extract a portion of a array list.

subList uses the Index like below..

fromIndex - Low endpoint (inclusive) of the subList.

toIndex - High endpoint (exclusive) of the subList.

List<String> sub\_List = list\_Strings.subList(0, 3);

\* Write a Java program to compare two array lists.

ArrayList<String> c1 = new ArrayList<String>();

c1.add("Red");

ArrayList<String> c2 = new ArrayList<String>();

c2.add("Red");

ArrayList<String> c3 = new ArrayList<String>();

for (String e : c1)

c3.add(c2.contains(e) ? "Yes" : "No");

System.out.println(c3);

\* Write a Java program to join two array lists.

// Let join above two list

ArrayList<String> a = new ArrayList<String>();

a.addAll(c1);

a.addAll(c2);

System.out.println("New array: " + a);

\* Write a Java program to clone an array list to another array list.

ArrayList<String> newc1 = (ArrayList<String>)c1.clone();

System.out.println("Cloned array list: " + newc1);

\* Write a Java program to empty an array list.

c1.removeAll(c1);

System.out.println("Array list after remove all elements "+c1);

System.out.println("Checking the above array list is empty or not! "+c1.isEmpty()); // true

\* Write a Java program to replace the second element of a ArrayList with the specified element.

String new\_color = "White";

colorList.set(1,new\_color);

\* Write a Java program to append the specified element to the end of a linked list.

LinkedList<String> l\_list = new LinkedList<String>();

l\_list.add("Red");

// print the linked list

System.out.println("The linked list: " + l\_list);

// Print the linked list

for (String element : l\_list) {

System.out.println(element);

}

// Print list elements in reverse order

l\_list.descendingIterator();

// use add() method to add values in the linked list

l\_list.add(1, "Yellow");

// Add an element at the beginning

l\_list.addFirst("White");

// Add an element at the end of list

l\_list.addLast("Pink");

\* Write a Java program to insert some elements at the specified position into a linked list.

// Add the collection in the second position of the existing linked list

l\_list.addAll(1, new\_l\_list);

// Find first element of the List

Object first\_element = l\_list.getFirst();

// Find last element of the List

Object last\_element = l\_list.getLast();

// Remove the element in third position from the said linked list

l\_list.remove(2);

// Remove the first element

Object firstElement = l\_list.removeFirst();

// Remove the last element

Object lastElement = l\_list.removeLast();

// Removing all the elements from the linked list

l\_list.clear();

//Swapping 1st(index 0) element(Red) with the 3rd(index 2) element (Black)

Collections.swap(l\_list, 0, 2);

// shuffle the elements of the Linkedlist

Collections.shuffle(l\_list);

// join above two LinkedList

LinkedList <String> a = new LinkedList <String> ();

a.addAll(c1);

a.addAll(c2);

// create an empty linked list then store the copy of above

LinkedList <String> newc1 = new LinkedList <String> ();

newc1 = (LinkedList)c1.clone();

\* remove and return the first element of a linked list.

System.out.println("Removed element: "+c1.pop());

// Retrieve but does not remove, the first element of a linked list

String x = c1.peekFirst();

// Checks whether the color "Orange" exists or not.

Boolean b = c1.contains("Orange");

//Convert a linked list to array list

List<String> list = new ArrayList<String>(linked\_list);

//comparison output in linked list

LinkedList<String> c3 = new LinkedList<String>();

for (String e : c1)

c3.add(c2.contains(e) ? "Yes" : "No");

System.out.println(c3);

// Replacing 2nd element with new value

c1.set(1, "Orange");

**HashSet Exercises**

// use add() method to add values in the hash set

HashSet<String> h\_set = new HashSet<String>();

h\_set.add("Red");

System.out.println("Original Hash Set: " + h\_set);

System.out.println("Size of the Hash Set: " + h\_set.size());

// Iterate the hash set

Iterator<String> itr = h\_set.iterator();

while (itr.hasNext()) {

System.out.println(itr.next());

}

OR

**for**(String c2 : c1) {

System.***out***.println(c2);

}

// Remove all elements

h\_set.removeAll(h\_set);

HashSet <String> new\_h\_set = new HashSet <String> ();

new\_h\_set = (HashSet)h\_set.clone();

// Create an Array then convert HashSet to Array

String[] new\_array = new String[h\_set.size()];

h\_set.toArray(new\_array);

// Creat a TreeSet of HashSet elements

Set<String> tree\_set = new TreeSet<String>(h\_set);

// Create a List from HashSet elements

List<String> list = new ArrayList<String>(h\_set);

//comparison output in hash set

HashSet<String>result\_set = new HashSet<String>();

for(String element : h\_set){

System.out.println(h\_set2.contains(element) ? "Yes" : "No");

}

Write a Java program to compare two sets and retain elements which are same on both sets.

h\_set1.retainAll(h\_set2);

// and the set becomes empty.

h\_set.clear();

**TreeSet Exercises**

TreeSet<String> tree\_set = new TreeSet<String>();

tree\_set.add("Red");

System.out.println(tree\_set);

// Print the tree list

for (String element : tree\_set) {

System.out.println(element);

}

// adding treetwo to treeone

tree\_set1.addAll(tree\_set2);

// reverse the order of the elements

Iterator it = t\_set.descendingIterator();

// Find first element of the tree set

Object first\_element = tree\_set.first();

// Find last element of the tree set

Object last\_element = tree\_set.last();

// clone a tree set list to another tree set

TreeSet<String> new\_t\_set = (TreeSet<String>)t\_set.clone();

**PriorityQueue Exercises**

PriorityQueue<String> queue=new PriorityQueue<String>();

queue.add("Red");

queue.add("Green");

queue.add("Orange");

queue.add("White");

queue.add("Black");

System.out.println(queue);

Output: [Black, Green, Orange, White, Red]

// adding queue2 to queue1

queue1.addAll(queue2);

//Convert a Queue list to array list

List<String> array\_list = new ArrayList<String>(pq1);

//Convert Priority Queue to a string representation

String str1;

str1 = pq1.toString();

**HashMap Exercises**

HashMap<Integer,String> hash\_map= new HashMap<Integer,String>();

hash\_map.put(1, "Red");

hash\_map.put(2, "Green");

for(Map.Entry x : hash\_map.entrySet()){

System.out.println(x.getKey()+" "+x.getValue());

}

**for**(**int** k : hash\_map.keySet()) {

System.***out***.println(k);

}

**for**(String v : hash\_map.values()) {

System.***out***.println(v);

}

// put all values in secondmap

hash\_map2.putAll(hash\_map1);

System.***out***.println(“hash\_map2 : ”+ hash\_map2);

// get value of key 3

String val = hash\_map.get(3);

// Removing all the elements from the linked map

hash\_map.clear();

// check if map is empty

boolean result = hash\_map.isEmpty();

// create a shallow copy of a HashMap

HashMap<Integer,String> new\_hash\_map= new HashMap<Integer,String>();

new\_hash\_map = (HashMap)hash\_map.clone();

//chech hashMap contains the key or not

if (hash\_map.containsKey("Green")) {

//key exists

System.out.println("yes! - " + hash\_map.get("Green"));

} else {

//key does not exists

System.out.println("no!");

}

//chech hashMap contains the value or not

if (hash\_map.containsValue("Green")) {

//value exists

System.out.println("Yes! ");

} else {

//value does not exists

System.out.println("no!");

}

// create set view for the map

Set set = hash\_map.entrySet();

// check set values

System.out.println("Set values: " + set);

// get keyset value from map

Set keyset = hash\_map.keySet();

// checking collection view of the map

System.out.println("Collection view is: "+ hash\_map.values());

**Java OOP: Exercise**

**---------------------- w3resource.com/java-exercises ---------------------------**

**25.** Write a Java program to find common elements from three sorted (in non-decreasing order) arrays.

ArrayList<Integer> common = **new** ArrayList<Integer>();

**int** array1[] = {2, 4, 8};

**int** array2[] = {2, 3, 4, 8, 10, 16};

**int** array3[] = {4, 8, 14, 40};

**int** x = 0, y = 0, z = 0;

**while** (x < array1.length && y < array2.length && z < array3.length){

**if** (array1[x] == array2[y] && array2[y] == array3[z]){

common.add(array1[x]);

x++;

y++;

z++;

}

**else** **if** (array1[x] < array2[y])

x++;

**else** **if** (array2[y] < array3[z])

y++;

**else**

z++;

}

System.***out***.println(common);

-------------------------------------------------------------

**26.** Write a Java program to move all 0's to the end of an array. Maintain the relative order of the other (non-zero) array elements.

**int** nums[] = {0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1};

**int** left = 0;

**int** right = nums.length-1;

Arrays.*sort*(nums);

**while**(left<right) {

**if**(nums[left]==1 && nums[right]==0) {

nums[left]=0;

nums[right]=1;

left++;

right--;

}

**else** **if**(nums[left]==0 && nums[right]==1) {

left++;

right--;

}

**else** **if**(nums[left]==1 && nums[right]==1) {

right--;

}

**else** **if**(nums[left]==0 && nums[right]==0) {

left++;

}

}

System.***out***.println(Arrays.*toString*(nums));

**-------------------------------------------------**

**class** Student{

**private** String name;

**private** **int** age;

**private** **int** id;

Student(String name, **int** age, **int** id){

**this**.name = name;

**this**.age = age;

**this**.id = id;

}

}

**class** work{

**public** **void** test() {

Student s1 = **new** Student("john", 34, 0021);

Student s2 = **new** Student("tom", 35, 0022);

ArrayList<Student> list = **new** ArrayList<>();

list.add(s1);

list.add(s2);

System.***out***.println(list);

**for**(Student student : list) {

System.***out***.println(student.name+" "+student.age+" "+student.id); }

}

}

**---------------------------OOPS----------------------------**

**7.** Write a Java program to create a class called "Bank" with a collection of accounts and methods to add and remove accounts, and to deposit and withdraw money. Also define a class called "Account" to maintain account details of a particular customer.

//Account.java

**public** **class** Account {

**private** String name;

**private** String accountNumber;

**private** **double** balance;

**public** Account(String name, String accountNumber, **double** balance) {

**this**.name = name;

**this**.accountNumber = accountNumber;

**this**.balance = balance;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getAccountNumber() {

**return** accountNumber;

}

**public** **void** setAccountNumber(String accountNumber) {

**this**.accountNumber = accountNumber;

}

**public** **double** getBalance() {

**return** balance;

}

**public** **void** setBalance(**double** balance) {

**this**.balance = balance;

}

**public** **void** deposit(**double** amount) {

balance += amount;

}

**public** **void** withdraw(**double** amount) {

balance -= amount;

}

**public** String getAccountInfo() {

**return** "Name: " + name + ", Account Number: " + accountNumber + ", Balance: " + balance;

}

}

--------------

//Bank.java

**import** java.util.ArrayList;

**public** **class** Bank {

**private** ArrayList < Account > accounts;

**public** Bank() {

accounts = **new** ArrayList < Account > ();

}

**public** **void** addAccount(Account account) {

accounts.add(account);

}

**public** **void** removeAccount(Account account) {

accounts.remove(account);

}

**public** **void** depositMoney(Account account, **double** amount) {

account.deposit(amount);

}

**public** **void** withdrawMoney(Account account, **double** amount) {

account.withdraw(amount);

}

**public** ArrayList < Account > getAccounts() {

**return** accounts;

}

}

------------------------

//Main.java

**import** java.util.ArrayList;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Bank bank = **new** Bank();

Account account1 = **new** Account("Peter Irmgard", "C0011", 5000);

Account account2 = **new** Account("Katja Ruedi", "C0121", 4500);

Account account3 = **new** Account("Marcella Gebhard", "C0222", 20000);

bank.addAccount(account1);

bank.addAccount(account2);

bank.addAccount(account3);

ArrayList < Account > accounts = bank.getAccounts();

**for** (Account account: accounts) {

System.***out***.println(account.getAccountInfo());

}

System.***out***.println("\nAfter depositing 1000 into account1:");

bank.depositMoney(account1, 1000);

System.***out***.println(account1.getAccountInfo());

System.***out***.println("No transaction in account2:");

System.***out***.println(account2.getAccountInfo());

System.***out***.println("After withdrawing 5000 from account3:");

bank.withdrawMoney(account3, 5000);

System.***out***.println(account3.getAccountInfo());

}

}

--------------------

Write a Java program to create a class called "Rectangle" with width and height attributes. Calculate the area and perimeter of the rectangle.

**Sample Solution:**

**Java Code:**

//Rectangle.java

public class Rectangle {

private double width;

private double height;

public Rectangle(double width, double height) {

this.width = width;

this.height = height;

}

public double getWidth() {

return width;

}

public void setWidth(double width) {

this.width = width;

}

public double getHeight() {

return height;

}

public void setHeight(double height) {

this.height = height;

}

public double getArea() {

return width \* height;

}

public double getPerimeter() {

return 2 \* (width + height);

}

}

//Main.java

public class Main {

public static void main(String[] args) {

Rectangle rectangle = new Rectangle(7, 12);

System.out.println("The area of the rectangle is " + rectangle.getArea());

System.out.println("The perimeter of the rectangle is " + rectangle.getPerimeter());

rectangle.setWidth(6);

rectangle.setHeight(12);

System.out.println("\nThe area of the rectangle is now " + rectangle.getArea());

System.out.println("The perimeter of the rectangle is now " + rectangle.getPerimeter());

}

}

Write a Java program to create a class called "Book" with attributes for title, author, and ISBN, and methods to add and remove books from a collection.

**Sample Solution:**

**Java Code:**

//Book.java

import java.util.ArrayList;

public class Book {

private String title;

private String author;

private String ISBN;

private static ArrayList < Book > bookCollection = new ArrayList < Book > ();

public Book(String title, String author, String ISBN) {

this.title = title;

this.author = author;

this.ISBN = ISBN;

}

public String get\_Title() {

return title;

}

public void set\_Title(String title) {

this.title = title;

}

public String get\_Author() {

return author;

}

public void set\_Author(String author) {

this.author = author;

}

public String get\_ISBN() {

return ISBN;

}

public void set\_ISBN(String ISBN) {

this.ISBN = ISBN;

}

public static void add\_Book(Book book) {

bookCollection.add(book);

}

public static void remove\_Book(Book book) {

bookCollection.remove(book);

}

public static ArrayList < Book > get\_BookCollection() {

return bookCollection;

}

}

//Main.java

import java.util.ArrayList;

public class Main {

public static void main(String[] args) {

Book book1 = new Book("The C Programming Language", "Dennis Ritchie, Brian Kernighan", "9780131101630");

Book book2 = new Book("An Introduction to Python", "Guido van Rossum", "9355423489");

Book.add\_Book(book1);

Book.add\_Book(book2);

ArrayList < Book > bookCollection = Book.get\_BookCollection();

System.out.println("List of books:");

for (Book book: bookCollection) {

System.out.println(book.get\_Title() + " by " + book.get\_Author() + ", ISBN: " + book.get\_ISBN());

}

Book.remove\_Book(book1);

System.out.println("\nAfter removing " + book1.get\_Title() + ":");

System.out.println("List of books:");

for (Book book: bookCollection) {

System.out.println(book.get\_Title() + " by " + book.get\_Author() + ", ISBN: " + book.get\_ISBN());

}

}

}

Write a Java program to create a class called "Bank" with a collection of accounts and methods to add and remove accounts, and to deposit and withdraw money. Also define a class called "Account" to maintain account details of a particular customer.

**Sample Solution:**

**Java Code:**

//Account.java

public class Account {

private String name;

private String accountNumber;

private double balance;

public Account(String name, String accountNumber, double balance) {

this.name = name;

this.accountNumber = accountNumber;

this.balance = balance;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAccountNumber() {

return accountNumber;

}

public void setAccountNumber(String accountNumber) {

this.accountNumber = accountNumber;

}

public double getBalance() {

return balance;

}

public void setBalance(double balance) {

this.balance = balance;

}

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) {

balance -= amount;

}

public String getAccountInfo() {

return "Name: " + name + ", Account Number: " + accountNumber + ", Balance: " + balance;

}

}

The above class has three private attributes: name, accountNumber and balance. There are several methods to deposit, withdraw, maintain balance in an individual account, print account details and more.

//Bank.java

import java.util.ArrayList;

public class Bank {

private ArrayList < Account > accounts;

public Bank() {

accounts = new ArrayList < Account > ();

}

public void addAccount(Account account) {

accounts.add(account);

}

public void removeAccount(Account account) {

accounts.remove(account);

}

public void depositMoney(Account account, double amount) {

account.deposit(amount);

}

public void withdrawMoney(Account account, double amount) {

account.withdraw(amount);

}

public ArrayList < Account > getAccounts() {

return accounts;

}

}

//Main.java

import java.util.ArrayList;

public class Main {

public static void main(String[] args) {

Bank bank = new Bank();

Account account1 = new Account("Peter Irmgard", "C0011", 5000);

Account account2 = new Account("Katja Ruedi", "C0121", 4500);

Account account3 = new Account("Marcella Gebhard", "C0222", 20000);

bank.addAccount(account1);

bank.addAccount(account2);

bank.addAccount(account3);

ArrayList < Account > accounts = bank.getAccounts();

for (Account account: accounts) {

System.out.println(account.getAccountInfo());

}

System.out.println("\nAfter depositing 1000 into account1:");

bank.depositMoney(account1, 1000);

System.out.println(account1.getAccountInfo());

System.out.println("No transaction in account2:");

System.out.println(account2.getAccountInfo());

System.out.println("After withdrawing 5000 from account3:");

bank.withdrawMoney(account3, 5000);

System.out.println(account3.getAccountInfo());

}

}