**LAB#05**

**Lab Task#01:**

1.Write a program for Selection sort that sorts an array containing numbers, prints all the sort values of array each followed by its location.

**Code: Output:**

import java.util.\*;

public class LAB05 {

public static void main(String[] args) {

Scanner no = new Scanner(System.in);

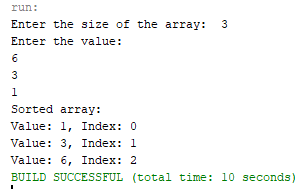
System.out.print("Enter the size of the array: ");

int n = no.nextInt();

int[] arr = new int[n];

System.out.println("Enter the value:");

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

 arr[i] = no.nextInt(); }

for (int i = 0; i < n - 1; i++) {

int Index = i;

for (int j = i + 1; j < n; j++) {

if (arr[j] < arr[Index]) {

Index = j; }

}

int temp = arr[Index];

arr[Index] = arr[i];

arr[i] = temp; }

System.out.println("Sorted array:");

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

System.out.println("Value: " + arr[i] + ", Index: " + i); } }

}

**Lab Task#02:**

Write a program that takes 10 numbers as input in an array. Sort the elements of array by using Bubble sort. Print each iteration of the sorting process.

**Code: Output:**

import java.util.\*;

public class LAB05 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

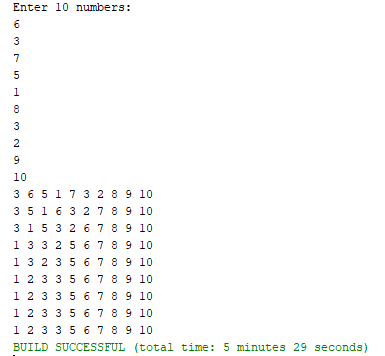
int[] a = new int[10];

System.out.println("Enter 10 numbers:");

for (int i = 0; i < 10; i++) {

a[i] = scanner.nextInt(); }

for (int i = 0; i < a.length - 1; i++) {

 for (int j = 0; j < a.length - i - 1; j++) {

if (a[j] > a[j + 1]) {

int temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp; }

}

for (int num : a) {

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

System.out.println(); }

}

}

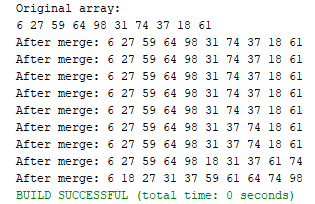
**Lab Task#03:**

3.Write a program that takes 10 random numbers in an array. Sort the elements of array by using Merge sort applying recursive technique. Print each iteration of the sorting process.

**Code: Output:**

import java.util.\*;

public class LAB05 {

 public static void mergeSort(int[] arr, int left, int right) {

if (left < right) {

int mid = (left + right) / 2;

mergeSort(arr, left, mid);

mergeSort(arr, mid + 1, right);

merge(arr, left, mid, right); }

}

public static void merge(int[] arr, int left, int mid, int right) {

int n1 = mid - left + 1, n2 = right - mid;

int[] leftArray = new int[n1], rightArray = new int[n2];

System.arraycopy(arr, left, leftArray, 0, n1);

System.arraycopy(arr, mid + 1, rightArray, 0, n2);

int i = 0, j = 0, k = left;

while (i < n1 && j < n2) {

if (leftArray[i] <= rightArray[j]) arr[k++] = leftArray[i++];

else arr[k++] = rightArray[j++]; }

while (i < n1) arr[k++] = leftArray[i++];

while (j < n2) arr[k++] = rightArray[j++];

System.out.print("After merge: ");

for (int num : arr) {

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

System.out.println(); }

public static void main(String[] args) {

Random r1 = new Random();

int[] arr = new int[10];

System.out.println("Original array:");

for (int i = 0; i < 10; i++) {

arr[i] = r1.nextInt(100);

System.out.print(arr[i] + " "); }

System.out.println();

mergeSort(arr, 0, arr.length - 1); }

}

**Home Task#01:**

1.Declare an array of size n to store account balances. Initialize with values 0 to 100000 and sort Account No’s according to highest balance values by using Quick sort.

**Code: Output:**

import java.util.\*;

public class LAB05 {

static class Account {

int accountNo, balance;

Account(int accountNo, int balance) {

this.accountNo = accountNo;

this.balance = balance; }

}

public static void quickSort(Account[] accounts, int low, int high) {

if (low < high) {

int pi = partition(accounts, low, high);

quickSort(accounts, low, pi - 1);

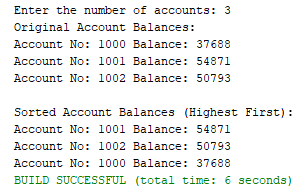
quickSort(accounts, pi + 1, high); }

}

public static int partition(Account[] accounts, int low, int high) {

int pivot = accounts[high].balance;

int i = low - 1;

 for (int j = low; j < high; j++) {

if (accounts[j].balance >= pivot) {

i++;

Account temp = accounts[i];

accounts[i] = accounts[j];

accounts[j] = temp; }

}

Account temp = accounts[i + 1];

accounts[i + 1] = accounts[high];

accounts[high] = temp;

return i + 1; }

public static void main(String[] args) {

Random r1 = new Random();

Scanner no = new Scanner(System.in);

System.out.print("Enter the number of accounts: ");

int n = no.nextInt();

Account[] accounts = new Account[n];

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

accounts[i] = new Account(1000 + i, r1.nextInt(100001)); }

System.out.println("Original Account Balances:");

for (Account account : accounts) {

System.out.println("Account No: " + account.accountNo + " Balance: " + account.balance); }

quickSort(accounts, 0, n - 1);

System.out.println("\nSorted Account Balances (Highest First):");

for (Account account : accounts) {

System.out.println("Account No: " + account.accountNo + " Balance: " + account.balance); }

}

}

**Home Task#02:**

2. Write a program which takes an unordered list of integers (or any other objects e.g. String), you have to rearrange the list in their natural order using merge sort

**Code:**

import java.util.\*;

public class LAB05 {

public static <T extends Comparable<T>> void mergeSort(List<T> list) {

if (list.size() <= 1) return;

int mid = list.size() / 2;

List<T> left = new ArrayList<>(list.subList(0, mid));

List<T> right = new ArrayList<>(list.subList(mid, list.size()));

mergeSort(left);

mergeSort(right);

merge(list, left, right); }

public static <T extends Comparable<T>> void merge(List<T> list, List<T> left, List<T> right) {

int i = 0, j = 0, k = 0;

while (i < left.size() && j < right.size()) {

list.set(k++, (left.get(i).compareTo(right.get(j)) <= 0) ? left.get(i++) : right.get(j++)); }

while (i < left.size()) list.set(k++, left.get(i++));

while (j < right.size()) list.set(k++, right.get(j++));

}

public static void main(String[] args) {

List<Integer> integers = new ArrayList<>(Arrays.asList(98, 4, 73, 30, 55, 65, 31));

System.out.println("Original List (Integers): " + integers);

mergeSort(integers);

System.out.println("Sorted List (Integers): " + integers);

List<String> strings = new ArrayList<>(Arrays.asList("Datsun", "Integra", "Silvia", "Impreza"));

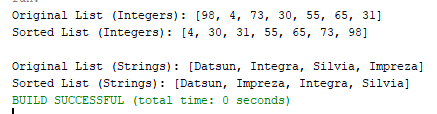
System.out.println("\nOriginal List (Strings): " + strings);

mergeSort(strings);

System.out.println("Sorted List (Strings): " + strings); }

}

**Output:**



**Home Task#03:**

3.You are given an unordered list of integers or strings. Write a program to Take this list as input. Sort it in naturalorder using Merge Sort. For integers, this means ascending order. For strings, this means alphabetical order. Print the sorted list.

**Code: Output:**

import java.util.\*;

import java.util.stream.Collectors;

public class LAB05 {

public static <T extends Comparable<T>> void mergeSort(List<T> list) {

if (list.size() <= 1) return;

int mid = list.size() / 2;

List<T> left = new ArrayList<>(list.subList(0, mid));

List<T> right = new ArrayList<>(list.subList(mid, list.size()));

mergeSort(left);

mergeSort(right);

merge(list, left, right); }

public static <T extends Comparable<T>> void merge(List<T> list, List<T> left, List<T> right) {

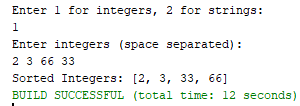
int i = 0, j = 0, k = 0;

while (i < left.size() && j < right.size())

list.set(k++, left.get(i).compareTo(right.get(j)) <= 0 ? left.get(i++) : right.get(j++));

while (i < left.size()) list.set(k++, left.get(i++));

while (j < right.size()) list.set(k++, right.get(j++)); }

 public static void main(String[] args) {

Scanner no = new Scanner(System.in);

System.out.println("Enter 1 for integers, 2 for strings:");

int choice = no.nextInt();

no.nextLine();

if (choice == 1) {

System.out.println("Enter integers (space separated):");

List<Integer> list = Arrays.asList(no.nextLine().split("\\s+")).stream()

.map(Integer::parseInt).collect(Collectors.toList());

mergeSort(list);

System.out.println("Sorted Integers: " + list);

} else if (choice == 2) {

System.out.println("Enter strings (space separated):");

List<String> list = Arrays.asList(no.nextLine().split("\\s+"));

mergeSort(list);

System.out.println("Sorted Strings: " + list);

} else {

System.out.println("Invalid choice"); }

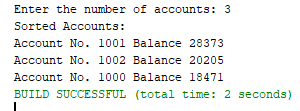
}

}

**Home Task#04:**

4.You are given a set of bank accounts, each with a unique account number and a balance. Write a Java program to Declare an array of size n to store account balances. Initialize each balance randomly with values between 0 and 100,000. Sort the accounts in descending order of their balances using Quick Sort. Print the sorted list in the format.

**Code: Output:**

import java.util.\*;

public class LAB05 {

static class Account {

int accountNumber;

int balance;

Account(int accountNumber, int balance) {

this.accountNumber = accountNumber;

this.balance = balance; }

public String toString() {

return "Account No. " + accountNumber + " Balance " + balance; }

}

public static void quickSort(Account[] accounts, int low, int high) {

if (low < high) {

int pivot = accounts[high].balance;

int i = low - 1;

for (int j = low; j < high; j++) {

if (accounts[j].balance > pivot) {

i++;

Account temp = accounts[i];

accounts[i] = accounts[j];

accounts[j] = temp; }

}

Account temp = accounts[i + 1];

accounts[i + 1] = accounts[high];

accounts[high] = temp;

int pivotIndex = i + 1;

quickSort(accounts, low, pivotIndex - 1);

quickSort(accounts, pivotIndex + 1, high); }

}

public static void main(String[] args) {

Scanner no = new Scanner(System.in);

System.out.print("Enter the number of accounts: ");

int n = no.nextInt();

Account[] accounts = new Account[n];

Random r1 = new Random();

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

accounts[i] = new Account(1000 + i, r1.nextInt(100001)); }

quickSort(accounts, 0, n - 1);

System.out.println("Sorted Accounts:");

for (Account account : accounts) {

System.out.println(account); }

}

}