**NAME:ROHAN NYATI**

**SAP ID:500075940**

**ROLL NO. : R177219148**

**BATCH-5(AI&ML)**

**Experiment-5**

1. **WAP for factorial computation.**

**CODE ->**

**public** **class** Factorial {

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

**int** i, fact = 1;

**int** number = 7;

**for**(i=1; i <= number; i++) {

fact = fact\*i;

}

System.***out***.println("Factorial of "+number+" is: "+fact);

}

}

**OUTPUT ->**



1. **WAP for displaying Fibonacci series by taking range from user.**

**CODE ->**

**import** java.util.Scanner;

**public** **class** Fibonacci {

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

**int** n, a = 0, b = 0, c = 1;

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter value of n:");

n = sc.nextInt();

System.***out***.print("Fibonacci Series:");

**for**(**int** i = 1; i <= n; i++)

{

a = b;

b = c;

c = a + b;

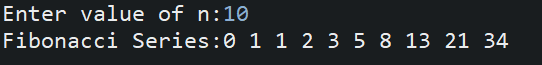
System.***out***.print(a+" ");

}

}

}

**OUTPUT ->**



1. **WAP for GCD calculation.**

**CODE ->**

**public** **class** GCD {

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

**int** x = 15, y = 17, gcd = 1;

**for**(**int** i = 1; i <= x && i <= y; i++)

{

**if**(x%i ==0 && y%i == 0)

gcd = i;

}

System.***out***.printf("GCD of %d and %d is: %d", x, y, gcd);

}

}

**OUTPUT ->**



1. **WAP to implement Bubble Sort, Insertion Sort, Merge Sort.**

**BUBBLE SORT:**

**CODE ->**

**public** **class** Bubble\_Sort {

**public** **static** **void** bubbleSort(**int**[] arr) {

**int** n = arr.length;

**int** temp = 0;

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

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

**if**(arr[j-1] > arr[j]) {

//swap elements

temp = arr[j-1];

arr[j-1] = arr[j];

arr[j] = temp;

}

}

}

}

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

**int** arr[] ={8, 24, 29, 15, 72, 50, 48, 65};

System.***out***.print("Before Bubble Sort -> ");

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

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

}

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

*bubbleSort*(arr); //sorting array elements using bubble sort

System.***out***.print("After Bubble Sort -> ");

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

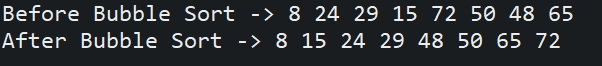
System.***out***.print(arr[i] + " ");

}

}

}

**OUTPUT ->**



**INSERTION SORT:**

**CODE ->**

**public** **class** Insertion\_Sort {

**public** **static** **void** insertionSort(**int** array[]) {

**int** n = array.length;

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

**int** key = array[j];

**int** i = j-1;

**while** ( (i > -1) && ( array [i] > key ) ) {

array [i+1] = array [i];

i--;

}

array[i+1] = key;

}

}

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

**int**[] arr1 = {7, 15, 10, 24, 29, 8, 55, 40};

System.***out***.print("Before Insertion Sort -> ");

**for**(**int** i:arr1){

System.***out***.print(i+" ");

}

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

*insertionSort*(arr1); //sorting array using insertion sort

System.***out***.print("After Insertion Sort -> ");

**for**(**int** i:arr1){

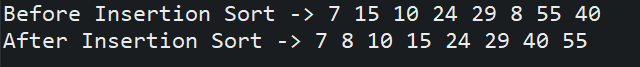
System.***out***.print(i+" ");

}

}

}

**OUTPUT ->**



**MERGE SORT:**

**CODE ->**

**public** **class** Merge\_Sort {

**public** **static** **void** merge(**int** arr[], **int** beg, **int** mid, **int** end)

{

**int** l = mid - beg + 1;

**int** r = end - mid;

**int** LeftArray[] = **new** **int** [l];

**int** RightArray[] = **new** **int** [r];

**for** (**int** i = 0; i < l; ++i)

LeftArray[i] = arr[beg + i];

**for** (**int** j = 0; j < r; ++j)

RightArray[j] = arr[mid + 1+ j];

**int** i = 0, j = 0;

**int** k = beg;

**while** (i < l && j < r)

{

**if** (LeftArray[i] <= RightArray[j])

{

arr[k] = LeftArray[i];

i++;

}

**else**

{

arr[k] = RightArray[j];

j++;

}

k++;

}

**while** (i < l)

{

arr[k] = LeftArray[i];

i++;

k++;

}

**while** (j < r)

{

arr[k] = RightArray[j];

j++;

k++;

}

}

**void** sort(**int** arr[], **int** beg, **int** end)

{

**if** (beg<end)

{

**int** mid = (beg+end)/2;

sort(arr, beg, mid);

sort(arr , mid+1, end);

*merge*(arr, beg, mid, end);

}

}

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

{

**int** arr[] = {12 , 20 , 47 , 29 , 54 , 60 , 7};

Merge\_Sort ob = **new** Merge\_Sort();

ob.sort(arr, 0, arr.length-1);

System.***out***.println("Sorted array -> ");

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

{

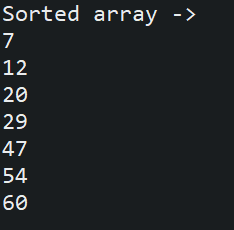
System.***out***.println(arr[i]);

}

}

}

**OUTPUT ->**



1. **WAP to implement Binary Search.**

**CODE ->**

**public** **class** Binary\_Search {

**public** **static** **int** binarySearch (**int** arr[], **int** first, **int** last, **int** key) {

**if** (last >= first)

{

**int** mid = first + (last - first)/2;

**if** (arr[mid] == key)

{

**return** mid;

}

**else** **if** (arr[mid] > key)

{

**return** *binarySearch*(arr, first, mid-1, key); //search in left subarray

}

**else**

{

**return** *binarySearch*(arr, mid+1, last, key); //search in right subarray

}

}

**return** -1;

}

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

**int** arr[] = {10, 20, 30, 40, 50};

**int** key = 30;

**int** last = arr.length - 1;

**int** result = *binarySearch*(arr, 0, last, key);

**if** (result == -1)

System.***out***.println("Element is not found!");

**else**

System.***out***.println("Element found at index -> "+result);

}

}

**OUTPUT ->**

