**DAA PRACTICAL**

**Name:Parth Borse Roll No.: 07**

**MERGE SORT PROGRAM**

#include<iostream>

using namespace std;

#define max 100

// Function prototypes

void merge\_sort(int arr[], int low, int up);

void merge\_s(int arr[], int temp[], int low1, int up1, int low2, int up2);

void copy\_s(int arr[], int temp[], int low, int up); // Function to copy elements from temporary array back to original array

int main() {

int i, n;

int arr[max];

cout << "Enter the size of array:" << endl;

cin >> n; // Input the size of the array

cout << "Enter array elements:" << endl;

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

cin >> arr[i];

}

merge\_sort(arr, 0, n - 1); // Call merge\_sort function to sort the array

cout << "Sorted list is:" << endl;

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

cout << arr[i] << " ";

}

cout << endl;

return 0;

}

// Function to perform merge sort

void merge\_sort(int arr[], int low, int up) {

int mid;

int temp[max];

if (low < up) {

mid = (low + up) / 2;

merge\_sort(arr, low, mid);

merge\_sort(arr, mid + 1, up);

merge\_s(arr, temp, low, mid, mid + 1, up); // Merge the sorted sublists

copy\_s(arr, temp, low, up);

}

}

// Function to merge two sublists

void merge\_s(int arr[], int temp[], int low1, int up1, int low2, int up2) {

int i = low1;

int j = low2;

int k = low1;

while ((i <= up1) && (j <= up2)) {

if (arr[i] <= arr[j]) {

temp[k++] = arr[i++];

} else {

temp[k++] = arr[j++];

}

}

while (i <= up1) {

temp[k++] = arr[i++];

}

while (j <= up2) { // Copy remaining elements from second sublist to merged sublist

temp[k++] = arr[j++];

}

}

// Function to copy elements from temporary array back to original array

void copy\_s(int arr[], int temp[], int low, int up) {

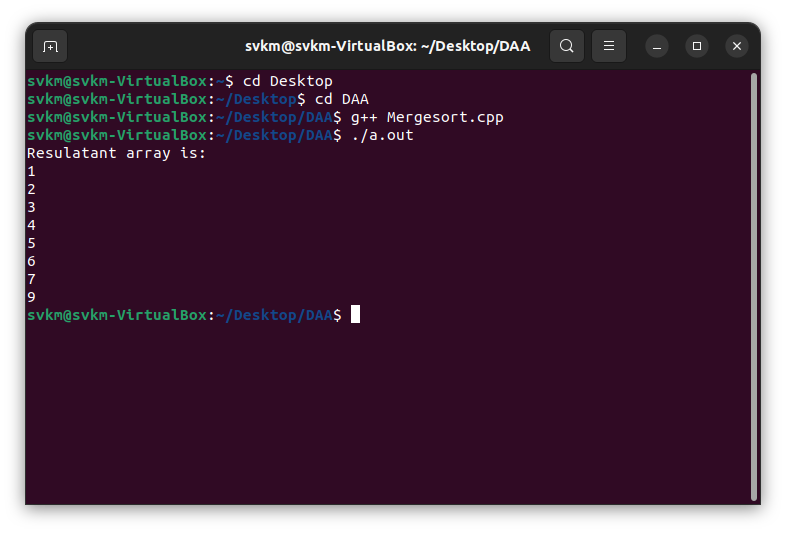
int i; // Loop variable

for (i = low; i <= up; i++) {

arr[i] = temp[i];

}

}

//OUTPUT : ****