Sorting Array Solutions

Q1. Write a program to sort an array in descending order using bubble sort. Input Array {3,5,1,6,0} Output Array: {6, 5, 3, 1, 0}

public class BubbleSortDescending {

public static void main(String[] args) {

// Initialize the array

int[] array = {3, 5, 1, 6, 0};

// Print the original array

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

printArray(array);

// Sort the array in descending order

bubbleSortDescending(array);

// Print the sorted array

System.out.println("Sorted array in descending order:");

printArray(array);

}

// Method to perform bubble sort in descending order

public static void bubbleSortDescending(int[] array) {

int n = array.length;

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

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

// Swap if the current element is less than the next element

if (array[j] < array[j + 1]) {

int temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

}

}

}

// Method to print the array

public static void printArray(int[] array) {

for (int num : array) {

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

}

System.out.println();

}

}

Q2. WAP to sort an array in descending order using selection sort

Input Array {3,5,1,6,0}

Output Array: {6, 5, 3, 1, 0}

public class SelectionSortDescending {

public static void main(String[] args) {

// Initialize the array

int[] array = {3, 5, 1, 6, 0};

// Print the original array

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

printArray(array);

// Sort the array in descending order

selectionSortDescending(array);

// Print the sorted array

System.out.println("Sorted array in descending order:");

printArray(array);

}

// Method to perform selection sort in descending order

public static void selectionSortDescending(int[] array) {

int n = array.length;

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

// Find the index of the maximum element in the remaining unsorted portion

int maxIndex = i;

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

if (array[j] > array[maxIndex]) {

maxIndex = j;

}

}

// Swap the found maximum element with the element at index i

if (maxIndex != i) {

int temp = array[i];

array[i] = array[maxIndex];

array[maxIndex] = temp;

}

}

}

// Method to print the array

public static void printArray(int[] array) {

for (int num : array) {

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

}

System.out.println();

}

}

Q3. WAP to sort an array in decreasing order using insertion sort Input Array {3,5,1,6,0}

Output Array: {6, 5, 3, 1, 0} public class InsertionSortDescending {

public static void main(String[] args) {

// Initialize the array

int[] array = {3, 5, 1, 6, 0};

// Print the original array

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

printArray(array);

// Sort the array in decreasing order

insertionSortDescending(array);

// Print the sorted array

System.out.println("Sorted array in decreasing order:");

printArray(array);

}

// Method to perform insertion sort in decreasing order

public static void insertionSortDescending(int[] array) {

int n = array.length;

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

int key = array[i];

int j = i - 1;

// Shift elements of the sorted portion that are less than the key

while (j >= 0 && array[j] < key) {

array[j + 1] = array[j];

j--;

}

array[j + 1] = key;

}

}

// Method to print the array

public static void printArray(int[] array) {

for (int num : array) {

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

}

System.out.println();

}

}

Q4. Find out how many pass would be required to sort the following array in decreasing order using bubble sort Input Array {3,5,1,6,0}

public class BubbleSortPasses {

public static void main(String[] args) {

// Initialize the array

int[] array = {3, 5, 1, 6, 0};

// Find the number of passes required to sort the array

int passes = bubbleSortPasses(array);

// Print the number of passes

System.out.println("Number of passes required: " + passes);

}

// Method to perform bubble sort and count the number of passes

public static int bubbleSortPasses(int[] array) {

int n = array.length;

int passes = 0;

boolean swapped;

// Repeat until no swaps are made

do {

swapped = false;

passes++;

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

// Swap if current element is less than the next element

if (array[i] < array[i + 1]) {

int temp = array[i];

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

array[i + 1] = temp;

swapped = true;

}

}

// Decrease the range for the next pass

n--;

} while (swapped);

return passes;

}

}

Q5. Find out the number of iterations to sort the array in descending order using selection sort.

Input Array {3,5,1,6,0}

public class SelectionSortIterations {

public static void main(String[] args) {

// Initialize the array

int[] array = {3, 5, 1, 6, 0};

// Find the number of iterations required to sort the array

int iterations = selectionSortIterations(array);

// Print the number of iterations

System.out.println("Number of iterations required: " + iterations);

}

// Method to perform selection sort and count the number of iterations

public static int selectionSortIterations(int[] array) {

int n = array.length;

int iterations = 0;

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

// Find the index of the maximum element in the remaining unsorted portion

int maxIndex = i;

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

if (array[j] > array[maxIndex]) {

maxIndex = j;

}

}

// Swap the found maximum element with the element at index i

if (maxIndex != i) {

int temp = array[i];

array[i] = array[maxIndex];

array[maxIndex] = temp;

}

// Increment iteration count

iterations++;

}

return iterations;

}

}