Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

LAB NO 5

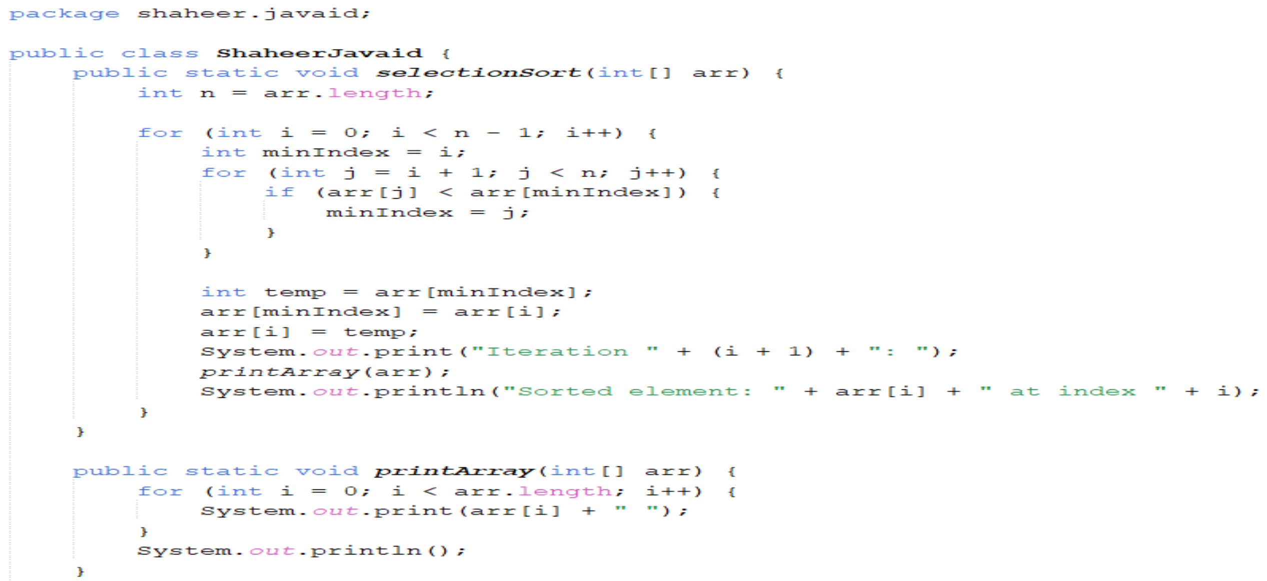
DATA STRUCTURES AND ALGORITHMS

OBJECTIVE: To sort a linear array using Selection Sort, Bubble Sort and Merge Sort.

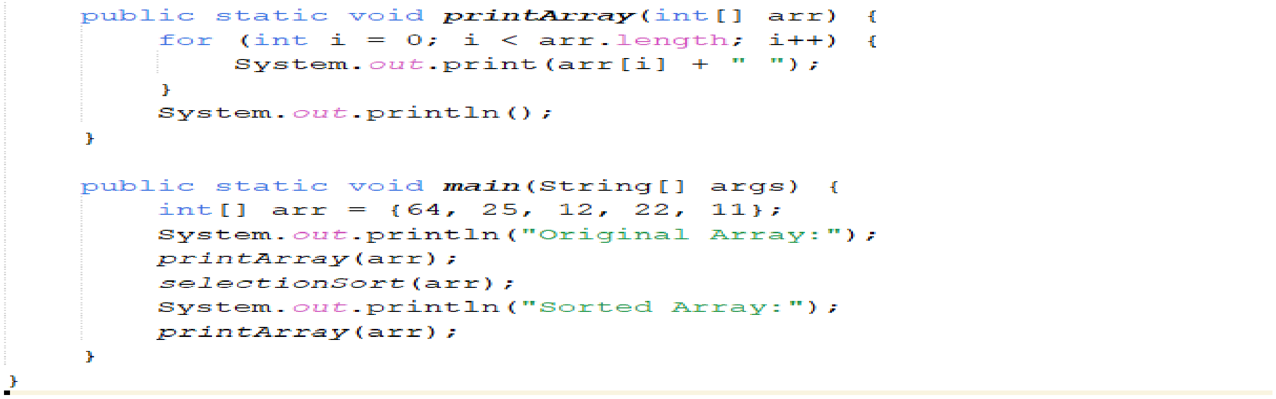
TASK NO 1:

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

INPUT:

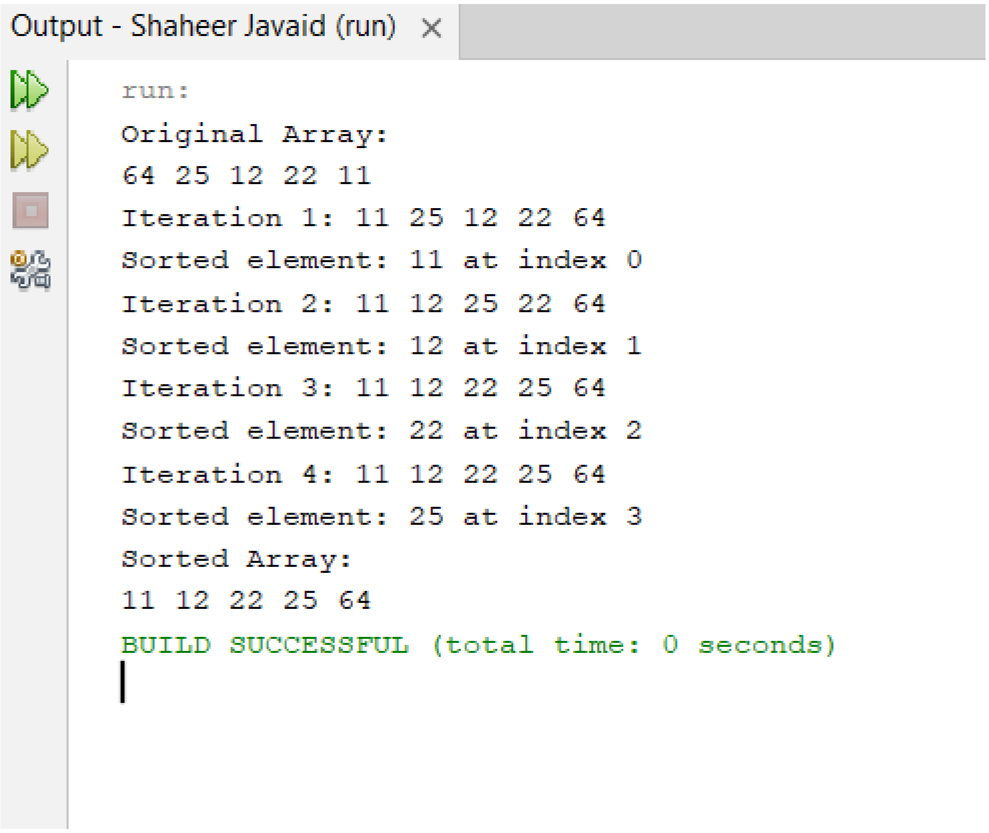


```
package shaheer.javaid;

public class ShaheerJavaid {
    public static void selectionSort(int[] arr) {
        int n = arr.length;

        for (int i = 0; i < n - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < n; j++) {
                if (arr[j] < arr[minIndex]) {
                    minIndex = j;
                }
            }

            int temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;
            System.out.print("Iteration " + (i + 1) + ": ");
            printArray(arr);
            System.out.println("Sorted element: " + arr[i] + " at index " + i);
        }
    }

    public static void printArray(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
```



```
    public static void printArray(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }

    public static void main(String[] args) {
        int[] arr = {64, 25, 12, 22, 11};
        System.out.println("Original Array:");
        printArray(arr);
        selectionSort(arr);
        System.out.println("Sorted Array:");
        printArray(arr);
    }
}
```

OUTPUT:

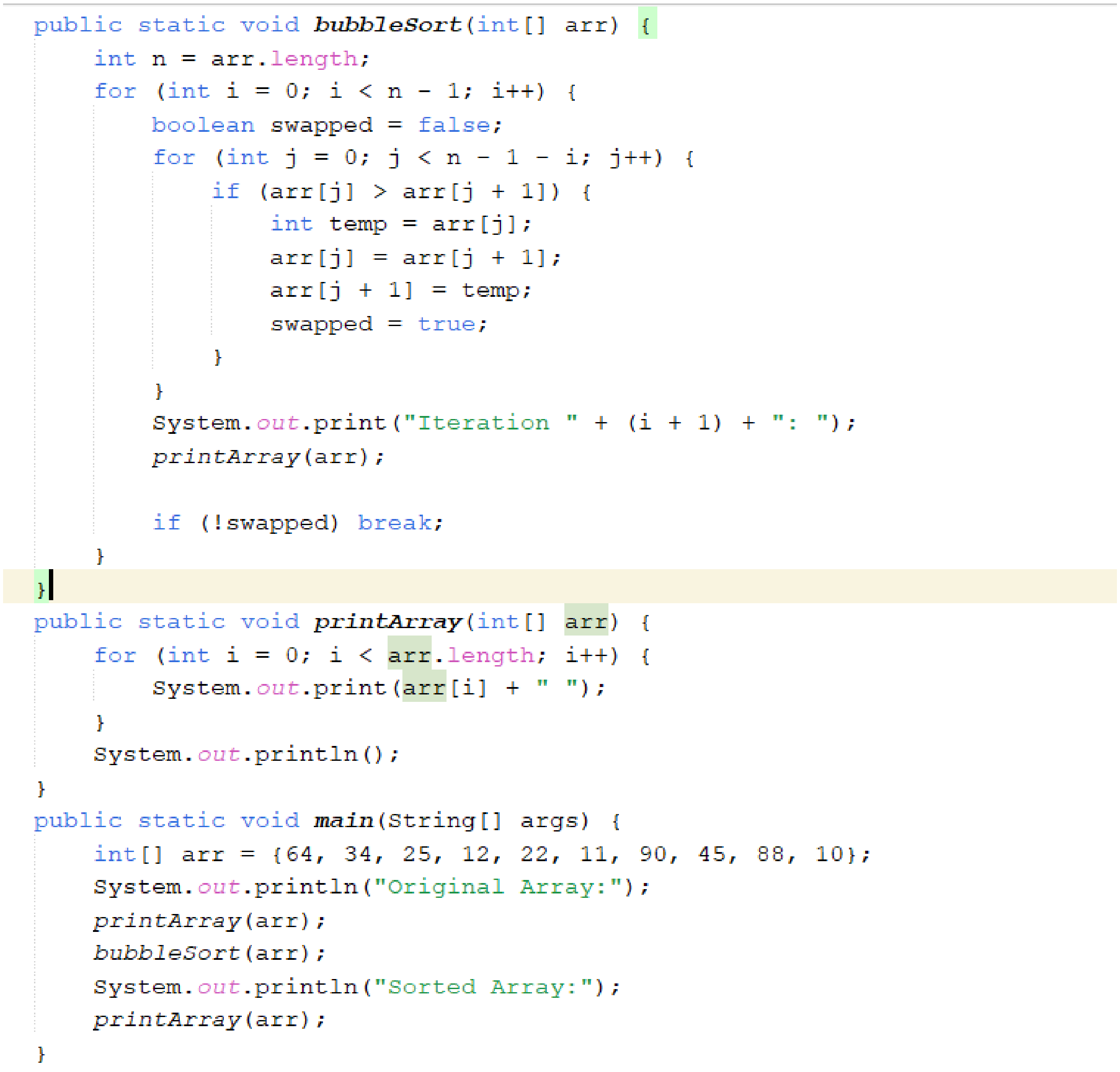


```
run:
Original Array:
64 25 12 22 11
Iteration 1: 11 25 12 22 64
Sorted element: 11 at index 0
Iteration 2: 11 12 25 22 64
Sorted element: 12 at index 1
Iteration 3: 11 12 22 25 64
Sorted element: 22 at index 2
Iteration 4: 11 12 22 25 64
Sorted element: 25 at index 3
Sorted Array:
11 12 22 25 64
BUILD SUCCESSFUL (total time: 0 seconds)
```

TASK NO 2:

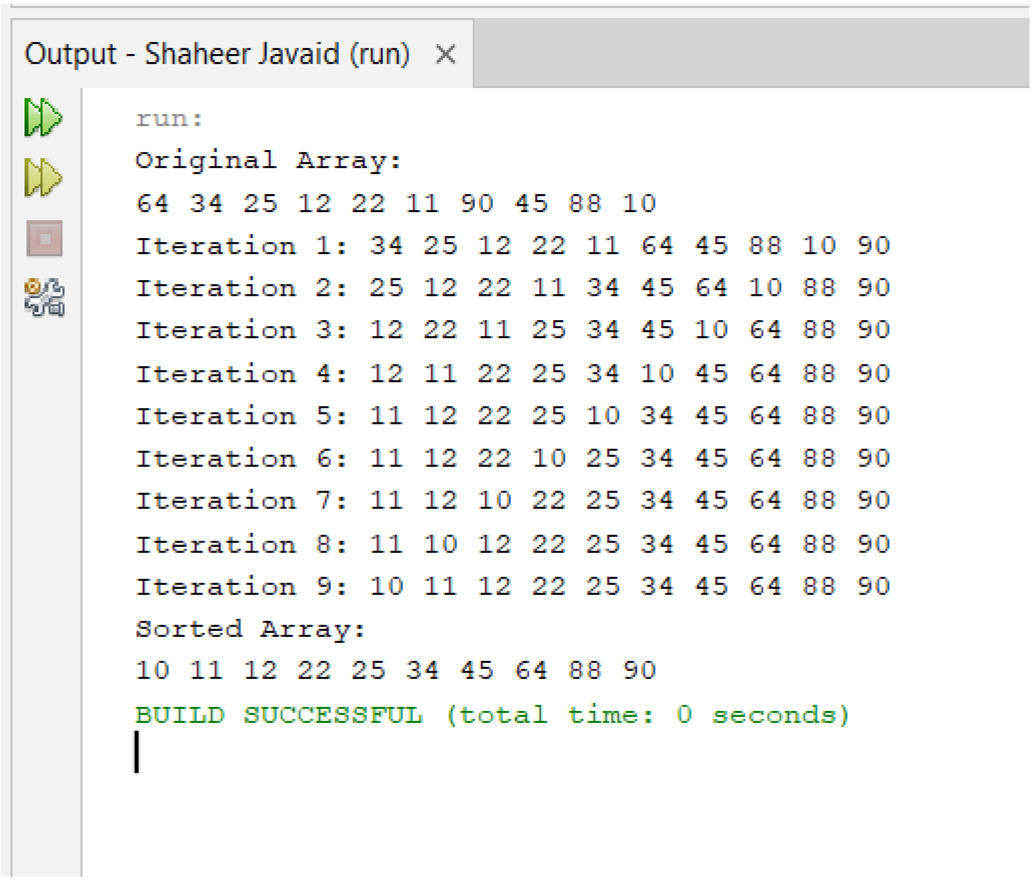
Write a program that takes 10 numbers as input in an array. Sort the elements of array by using Bubble sort. Print each iteration of the sorting process.

INPUT:



```java
public static void bubbleSort(int[] arr) {
    int n = arr.length;
    for (int i = 0; i < n - 1; i++) {
        boolean swapped = false;
        for (int j = 0; j < n - 1 - i; j++) {
            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
                swapped = true;
            }
        }
        System.out.print("Iteration " + (i + 1) + ": ");
        printArray(arr);

        if (!swapped) break;
    }
}
public static void printArray(int[] arr) {
    for (int i = 0; i < arr.length; i++) {
        System.out.print(arr[i] + " ");
    }
    System.out.println();
}
public static void main(String[] args) {
    int[] arr = {64, 34, 25, 12, 22, 11, 90, 45, 88, 10};
    System.out.println("Original Array:");
    printArray(arr);
    bubbleSort(arr);
    System.out.println("Sorted Array:");
    printArray(arr);
}
```

OUTPUT:

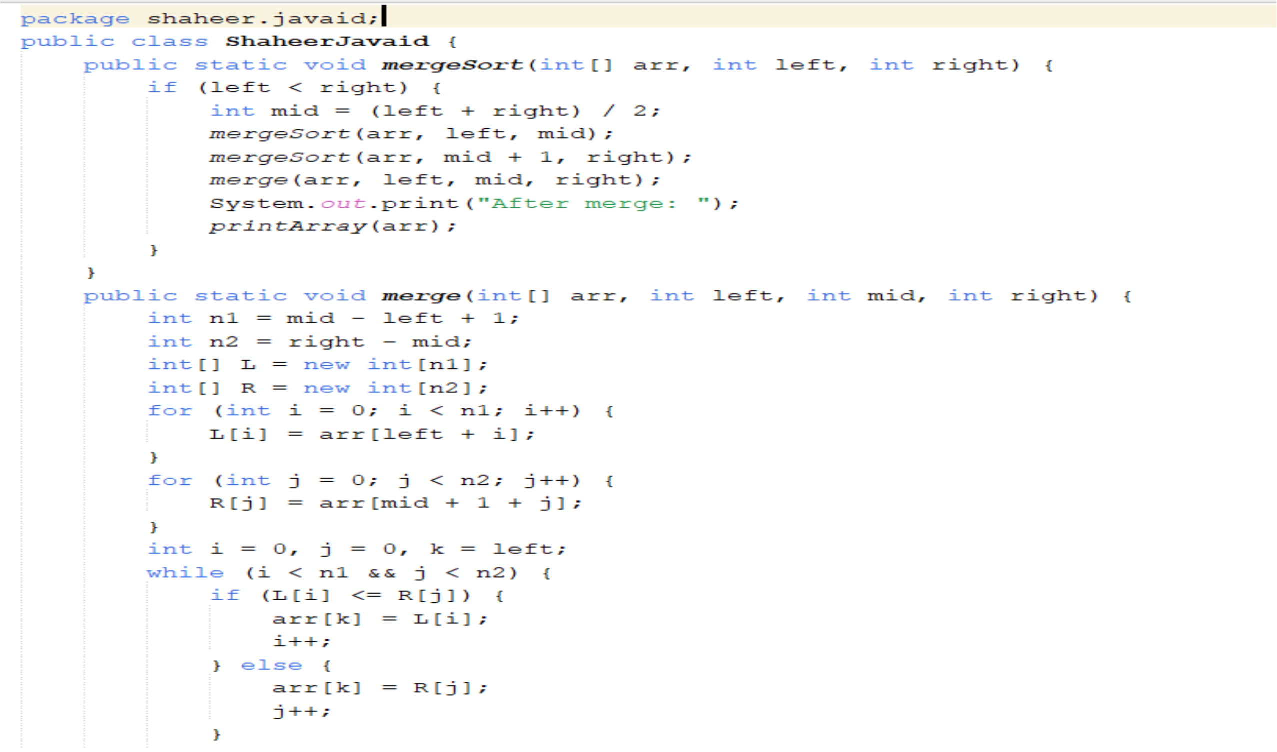


```
run:
Original Array:
64 34 25 12 22 11 90 45 88 10
Iteration 1: 34 25 12 22 11 64 45 88 10 90
Iteration 2: 25 12 22 11 34 45 64 10 88 90
Iteration 3: 12 22 11 25 34 45 10 64 88 90
Iteration 4: 12 11 22 25 34 10 45 64 88 90
Iteration 5: 11 12 22 25 10 34 45 64 88 90
Iteration 6: 11 12 22 10 25 34 45 64 88 90
Iteration 7: 11 12 10 22 25 34 45 64 88 90
Iteration 8: 11 10 12 22 25 34 45 64 88 90
Iteration 9: 10 11 12 22 25 34 45 64 88 90
Sorted Array:
10 11 12 22 25 34 45 64 88 90
BUILD SUCCESSFUL (total time: 0 seconds)
```

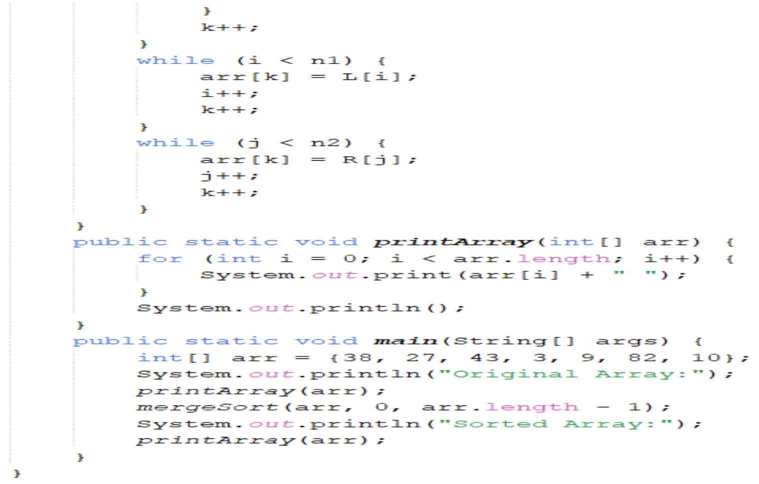
TASK NO 3:

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.

INPUT:

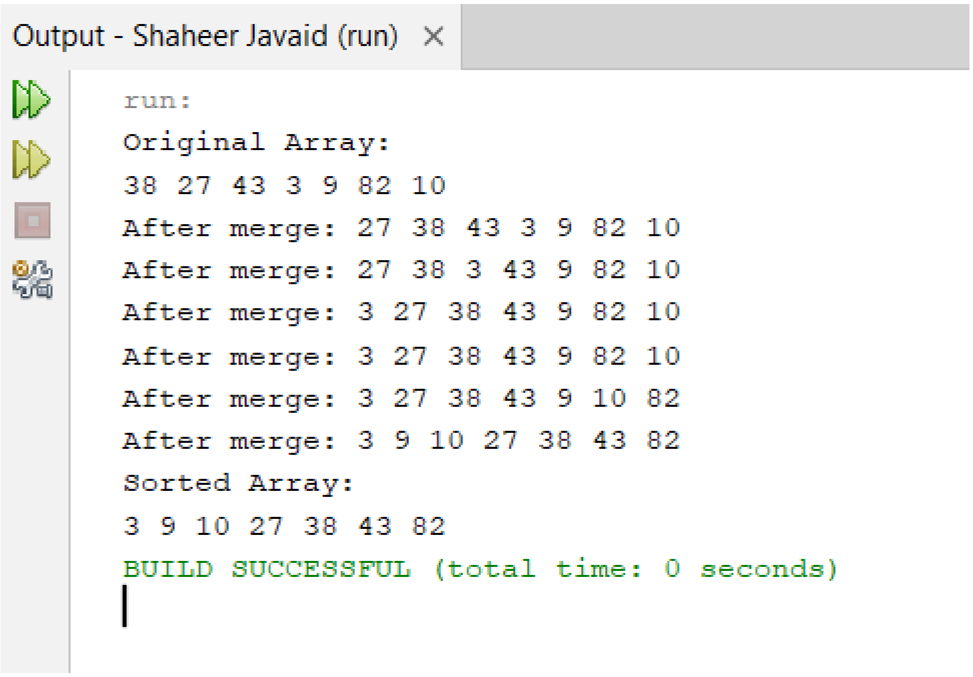


```
package shaheer.javaid;
public class ShaheerJavaid {
    public static void mergeSort(int[] arr, int left, int right) {
        if (left < right) {
            int mid = (left + right) / 2;
            mergeSort(arr, left, mid);
            mergeSort(arr, mid + 1, right);
            merge(arr, left, mid, right);
            System.out.print("After merge: ");
            printArray(arr);
        }
    }
    public static void merge(int[] arr, int left, int mid, int right) {
        int n1 = mid - left + 1;
        int n2 = right - mid;
        int[] L = new int[n1];
        int[] R = new int[n2];
        for (int i = 0; i < n1; i++) {
            L[i] = arr[left + i];
        }
        for (int j = 0; j < n2; j++) {
            R[j] = arr[mid + 1 + j];
        }
        int i = 0, j = 0, k = left;
        while (i < n1 && j < n2) {
            if (L[i] <= R[j]) {
                arr[k] = L[i];
                i++;
            } else {
                arr[k] = R[j];
                j++;
            }
```



```
            }
            k++;
        }
        while (i < n1) {
            arr[k] = L[i];
            i++;
            k++;
        }
        while (j < n2) {
            arr[k] = R[j];
            j++;
            k++;
        }
    }
    public static void printArray(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
    public static void main(String[] args) {
        int[] arr = {38, 27, 43, 3, 9, 82, 10};
        System.out.println("Original Array:");
        printArray(arr);
        mergeSort(arr, 0, arr.length - 1);
        System.out.println("Sorted Array:");
        printArray(arr);
    }
}
```

OUTPUT:



```
run:
Original Array:
38 27 43 3 9 82 10
After merge: 27 38 43 3 9 82 10
After merge: 27 38 3 43 9 82 10
After merge: 3 27 38 43 9 82 10
After merge: 3 27 38 43 9 82 10
After merge: 3 27 38 43 9 10 82
After merge: 3 9 10 27 38 43 82
Sorted Array:
3 9 10 27 38 43 82
BUILD SUCCESSFUL (total time: 0 seconds)
```

HOME TASKS

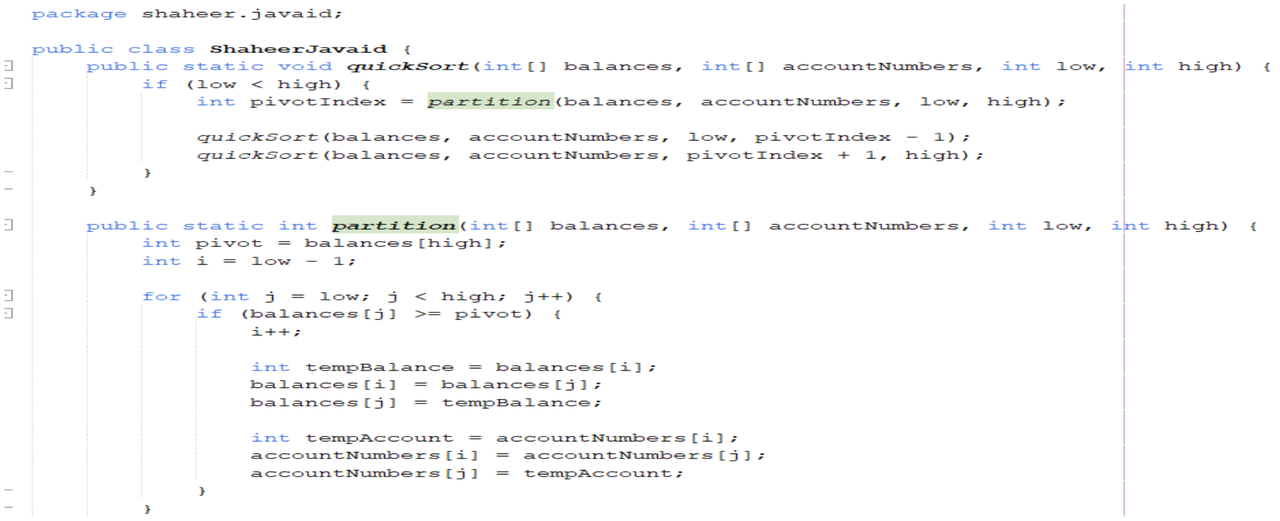
TASK NO 1:

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, For e.g.:

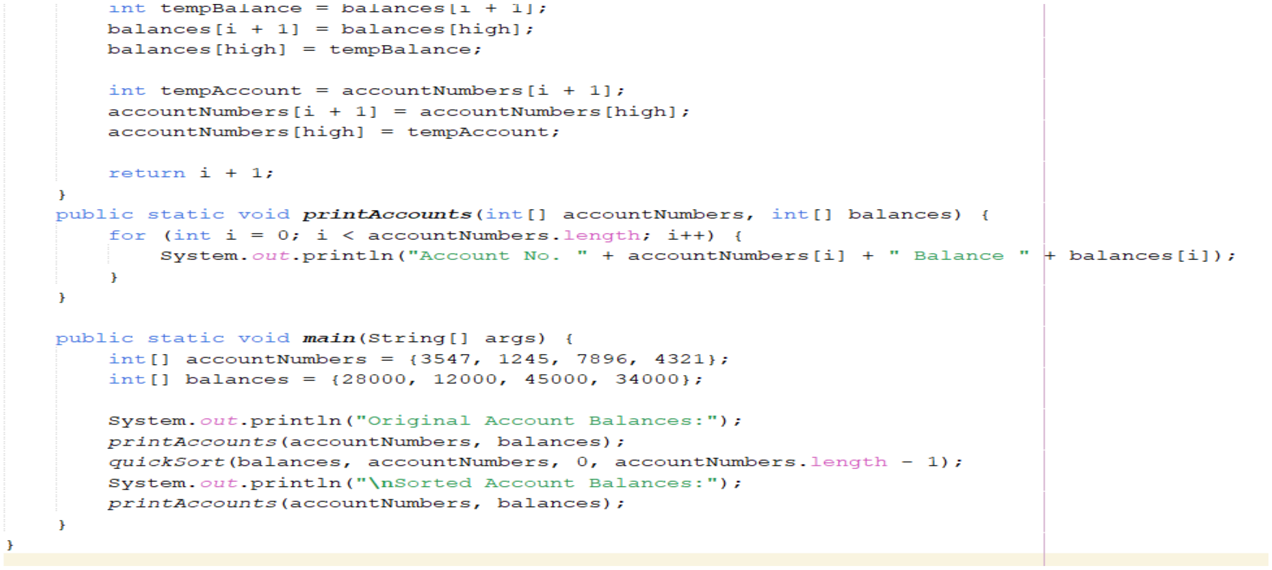
   Account No. 3547 Balance 28000

   Account No. 1245 Balance 12000

INPUT:

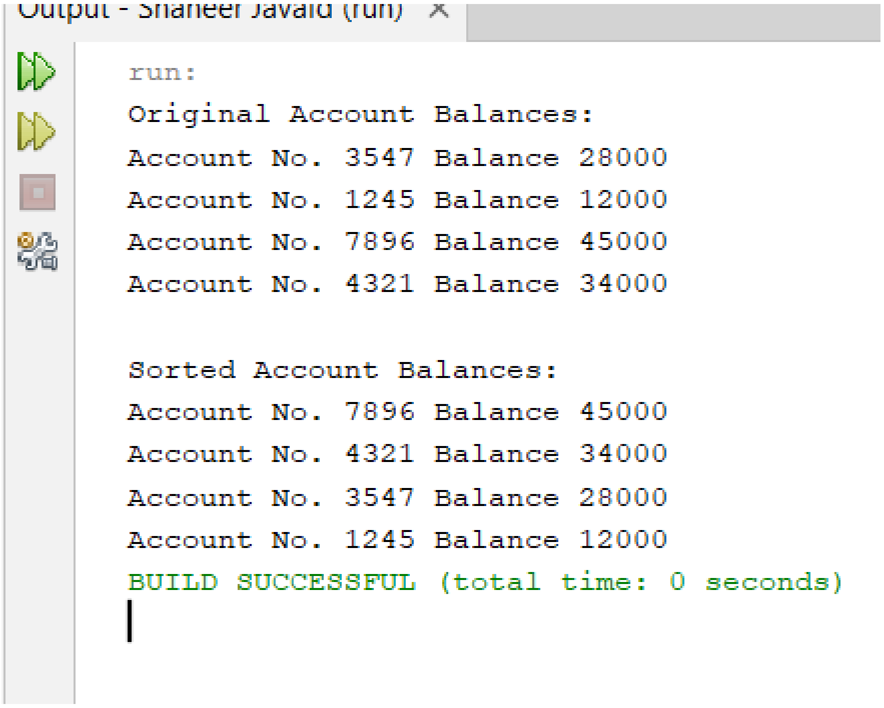


```java
package shaheer.javaid;

public class ShaheerJavaid {
    public static void quickSort(int[] balances, int[] accountNumbers, int low, int high) {
        if (low < high) {
            int pivotIndex = partition(balances, accountNumbers, low, high);

            quickSort(balances, accountNumbers, low, pivotIndex - 1);
            quickSort(balances, accountNumbers, pivotIndex + 1, high);
        }
    }

    public static int partition(int[] balances, int[] accountNumbers, int low, int high) {
        int pivot = balances[high];
        int i = low - 1;

        for (int j = low; j < high; j++) {
            if (balances[j] >= pivot) {
                i++;

                int tempBalance = balances[i];
                balances[i] = balances[j];
                balances[j] = tempBalance;

                int tempAccount = accountNumbers[i];
                accountNumbers[i] = accountNumbers[j];
                accountNumbers[j] = tempAccount;
            }
        }
```



```java
        int tempBalance = balances[i + 1];
        balances[i + 1] = balances[high];
        balances[high] = tempBalance;

        int tempAccount = accountNumbers[i + 1];
        accountNumbers[i + 1] = accountNumbers[high];
        accountNumbers[high] = tempAccount;

        return i + 1;
    }
    public static void printAccounts(int[] accountNumbers, int[] balances) {
        for (int i = 0; i < accountNumbers.length; i++) {
            System.out.println("Account No. " + accountNumbers[i] + " Balance " + balances[i]);
        }
    }

    public static void main(String[] args) {
        int[] accountNumbers = {3547, 1245, 7896, 4321};
        int[] balances = {28000, 12000, 45000, 34000};

        System.out.println("Original Account Balances:");
        printAccounts(accountNumbers, balances);
        quickSort(balances, accountNumbers, 0, accountNumbers.length - 1);
        System.out.println("\nSorted Account Balances:");
        printAccounts(accountNumbers, balances);
    }
}
```

OUTPUT:



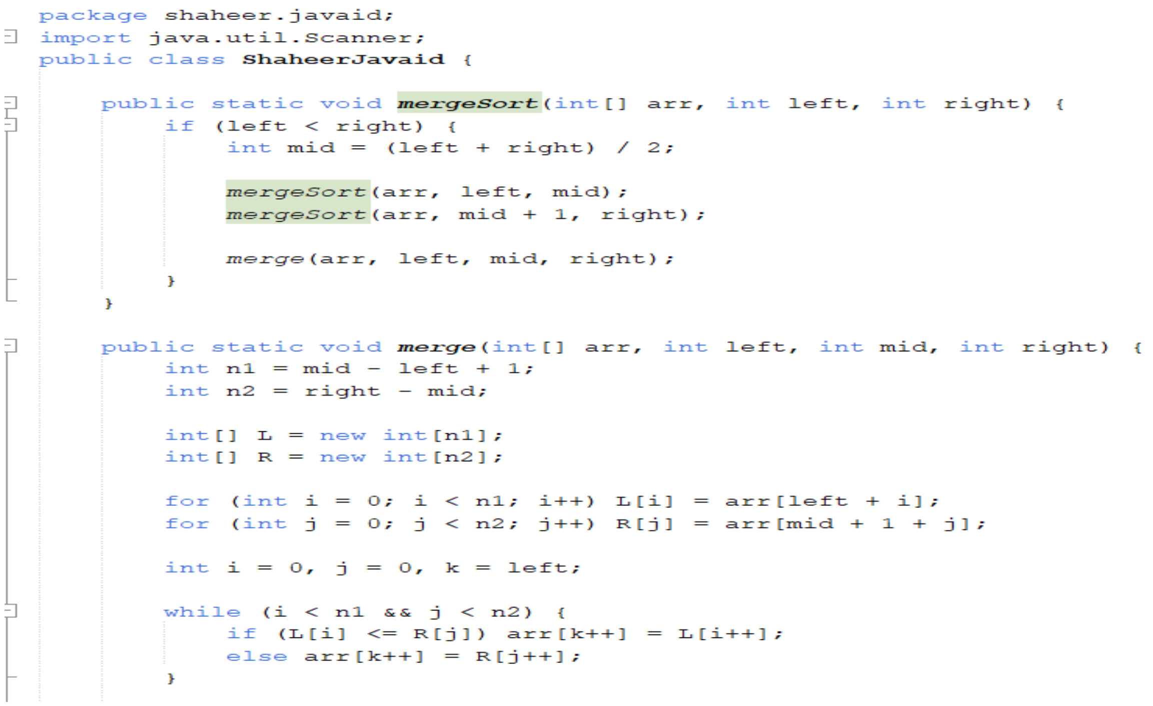
```
run:
Original Account Balances:
Account No. 3547 Balance 28000
Account No. 1245 Balance 12000
Account No. 7896 Balance 45000
Account No. 4321 Balance 34000

Sorted Account Balances:
Account No. 7896 Balance 45000
Account No. 4321 Balance 34000
Account No. 3547 Balance 28000
Account No. 1245 Balance 12000
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

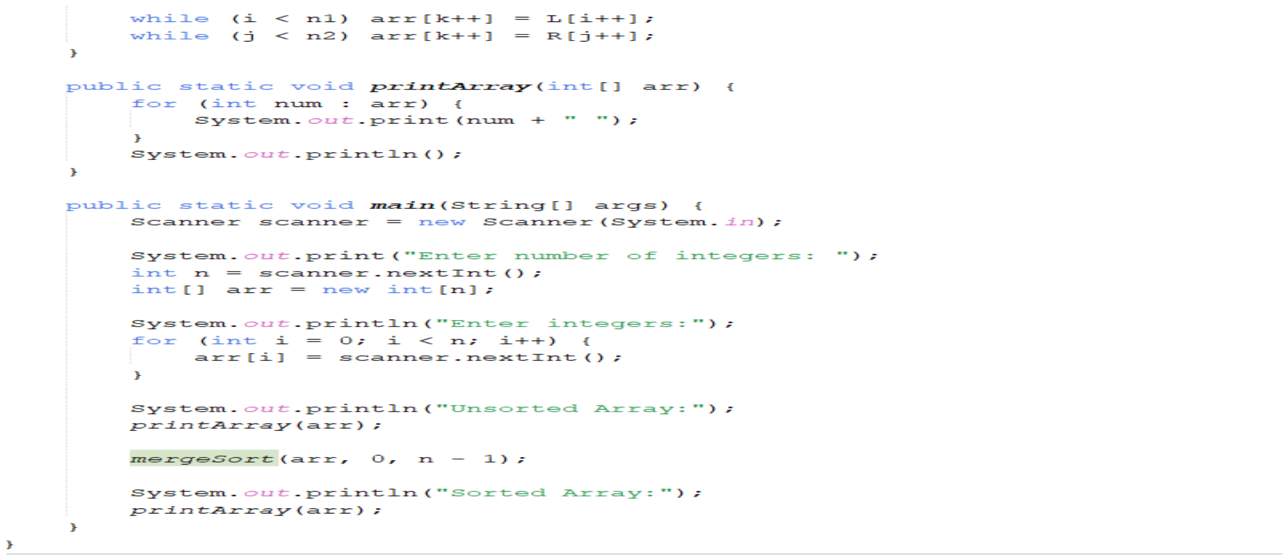
TASK NO 2

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

INPUT:

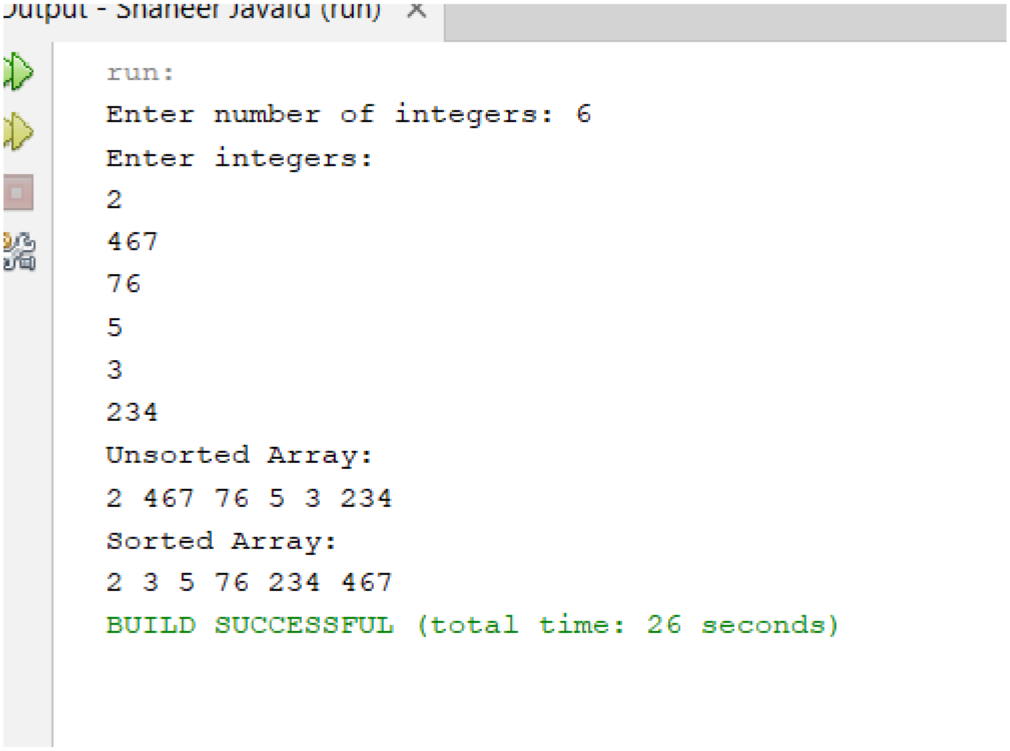


```java
package shaheer.javaid;
import java.util.Scanner;
public class ShaheerJavaid {

    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;
        int n2 = right - mid;

        int[] L = new int[n1];
        int[] R = new int[n2];

        for (int i = 0; i < n1; i++) L[i] = arr[left + i];
        for (int j = 0; j < n2; j++) R[j] = arr[mid + 1 + j];

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

        while (i < n1 && j < n2) {
            if (L[i] <= R[j]) arr[k++] = L[i++];
            else arr[k++] = R[j++];
        }
```



```java
        while (i < n1) arr[k++] = L[i++];
        while (j < n2) arr[k++] = R[j++];
    }
    public static void printArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter number of integers: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter integers:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        System.out.println("Unsorted Array:");
        printArray(arr);

        mergeSort(arr, 0, n - 1);

        System.out.println("Sorted Array:");
        printArray(arr);
    }
}
```

OUTPUT:



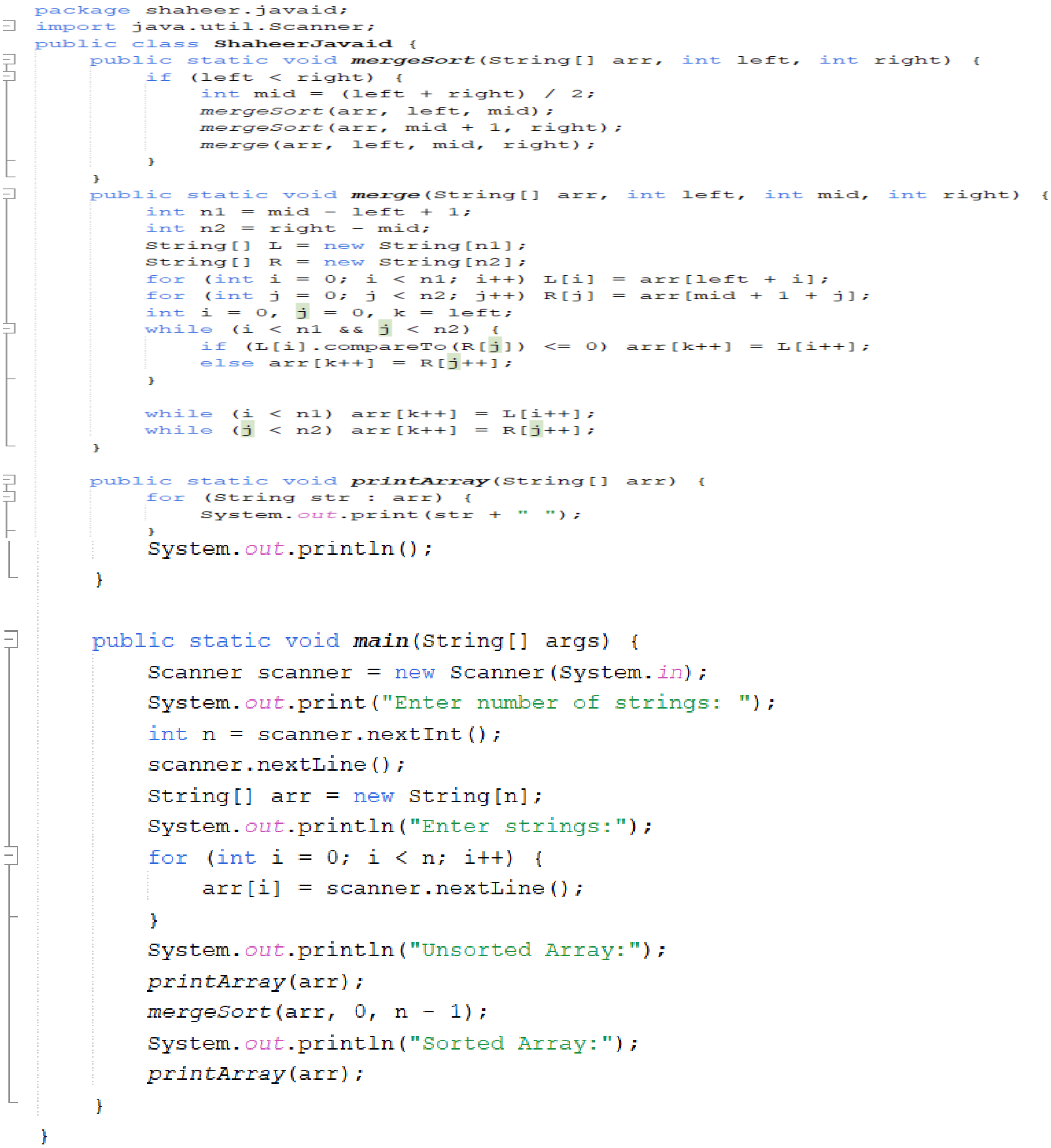
```
run:
Enter number of integers: 6
Enter integers:
2
467
76
5
3
234
Unsorted Array:
2 467 76 5 3 234
Sorted Array:
2 3 5 76 234 467
BUILD SUCCESSFUL (total time: 26 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 3:

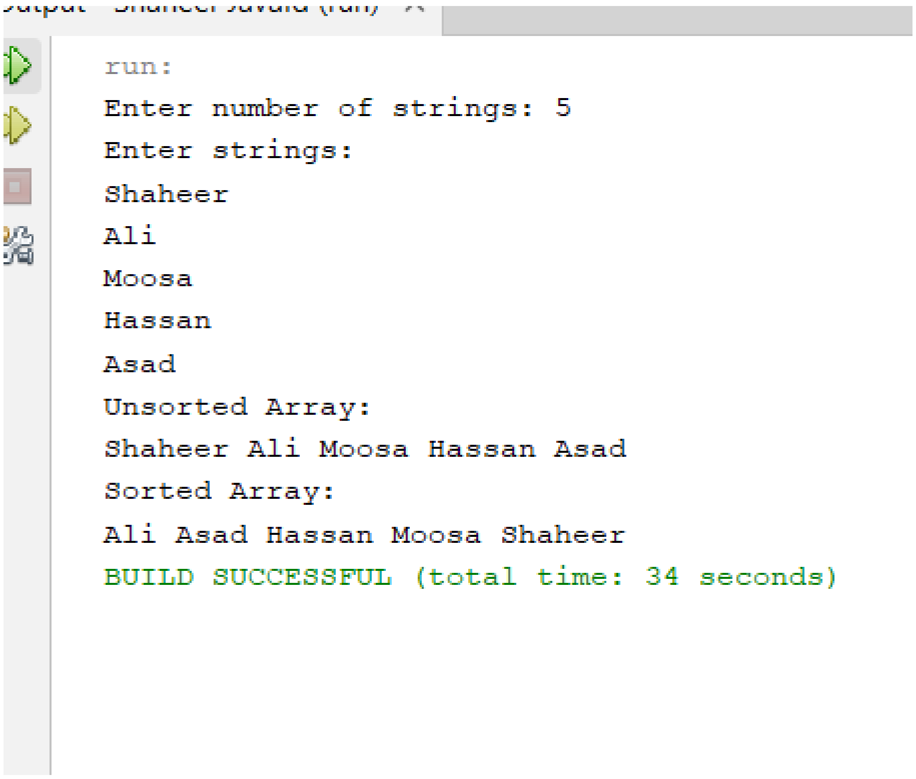
You are given an unordered list of integers or strings. Write a program to Take this list as input. Sort it in **natural order** using Merge Sort. For integers, this means ascending order. For strings, this means alphabetical order. Print the sorted list.

INPUT:



```
package shaheer.javaid;
import java.util.Scanner;
public class ShaheerJavaid {
    public static void mergeSort(String[] 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(String[] arr, int left, int mid, int right) {
        int n1 = mid - left + 1;
        int n2 = right - mid;
        String[] L = new String[n1];
        String[] R = new String[n2];
        for (int i = 0; i < n1; i++) L[i] = arr[left + i];
        for (int j = 0; j < n2; j++) R[j] = arr[mid + 1 + j];
        int i = 0, j = 0, k = left;
        while (i < n1 && j < n2) {
            if (L[i].compareTo(R[j]) <= 0) arr[k++] = L[i++];
            else arr[k++] = R[j++];
        }

        while (i < n1) arr[k++] = L[i++];
        while (j < n2) arr[k++] = R[j++];
    }

    public static void printArray(String[] arr) {
        for (String str : arr) {
            System.out.print(str + " ");
        }
        System.out.println();
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of strings: ");
        int n = scanner.nextInt();
        scanner.nextLine();
        String[] arr = new String[n];
        System.out.println("Enter strings:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextLine();
        }
        System.out.println("Unsorted Array:");
        printArray(arr);
        mergeSort(arr, 0, n - 1);
        System.out.println("Sorted Array:");
        printArray(arr);
    }
}
```

OUTPUT:

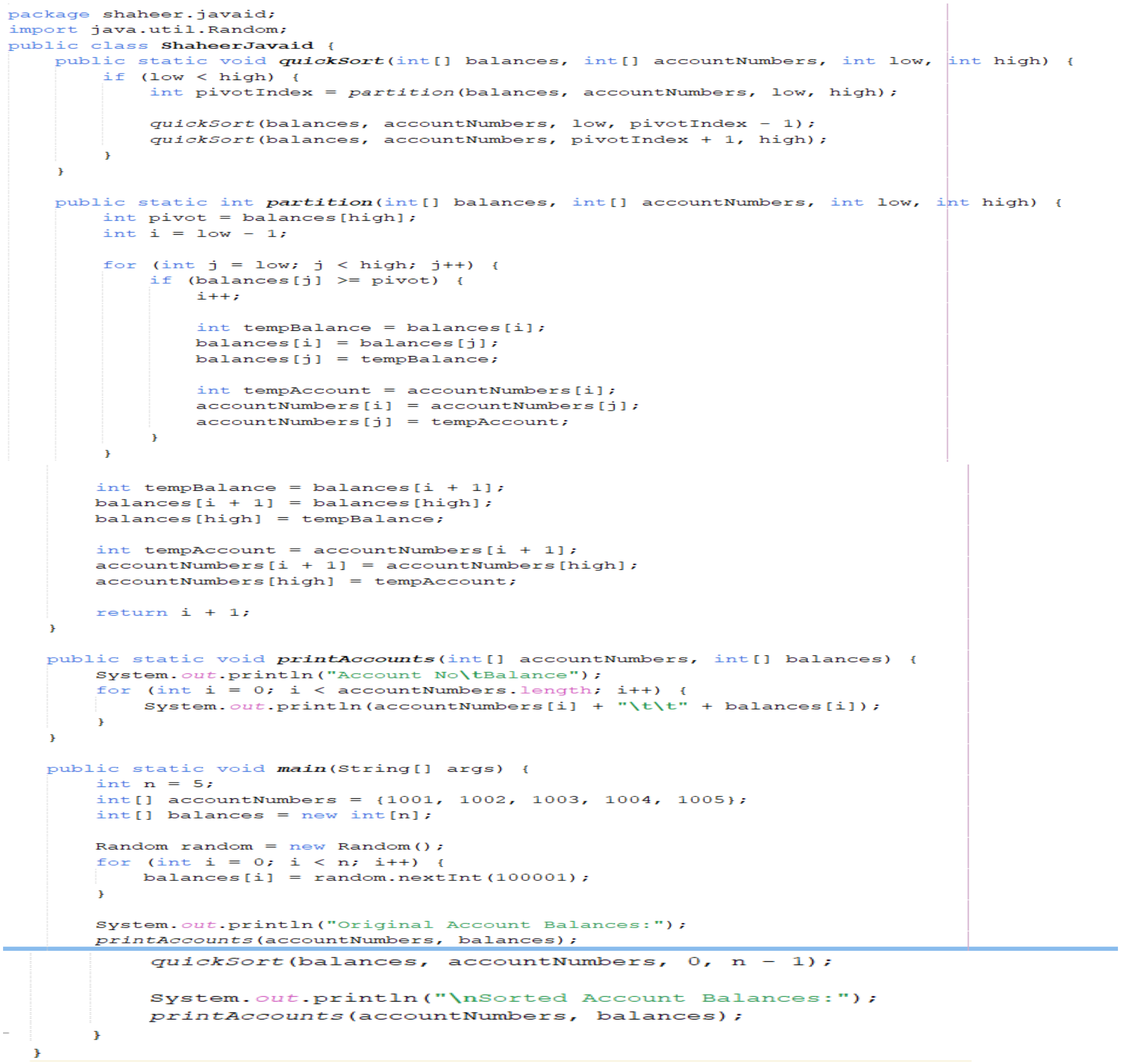


```
run:
Enter number of strings: 5
Enter strings:
Shaheer
Ali
Moosa
Hassan
Asad
Unsorted Array:
Shaheer Ali Moosa Hassan Asad
Sorted Array:
Ali Asad Hassan Moosa Shaheer
BUILD SUCCESSFUL (total time: 34 seconds)
```

TASK NO 4:

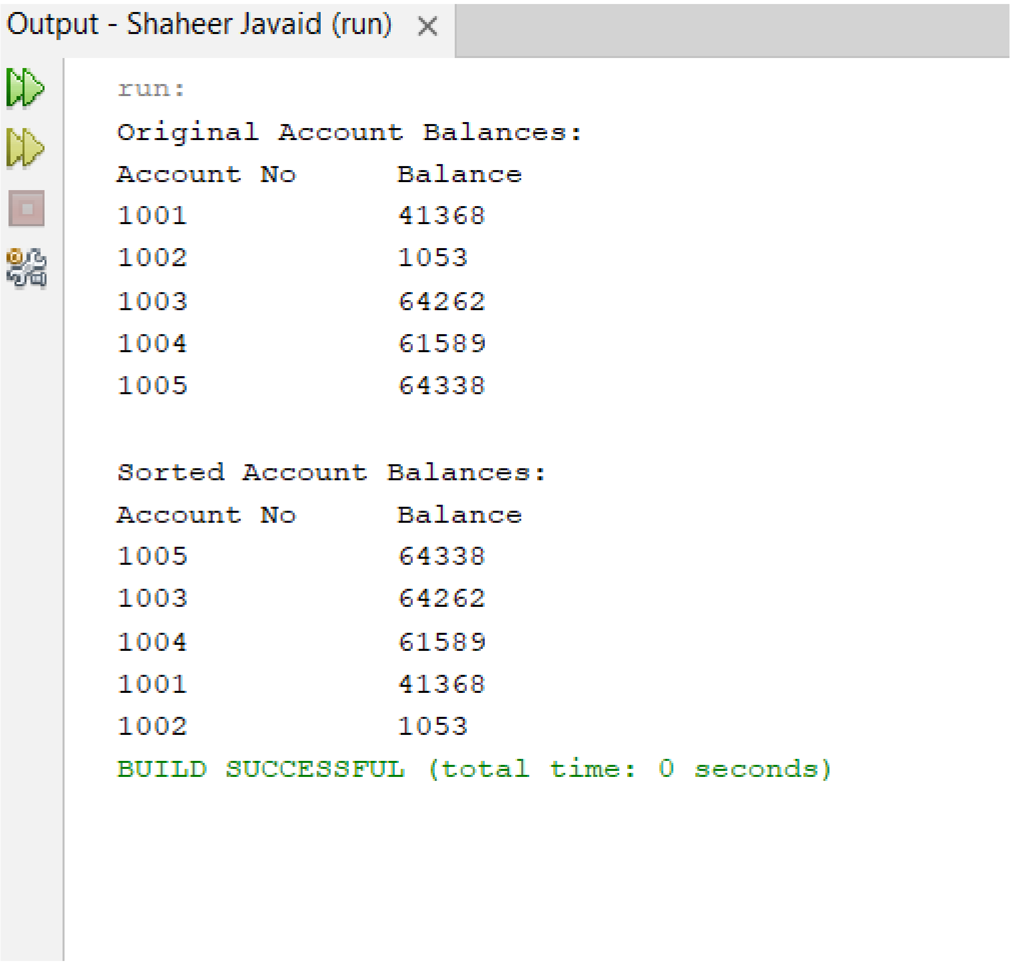
You are given a set of bank accounts, each with a unique account number and a balance. Write a Java program to Declare an array of size n to store account balances. Initialize each balance randomly with values between 0 and 100,000. Sort the accounts in **descending order** of their balances using Quick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

INPUT:



```java
package shaheer.javaid;
import java.util.Random;
public class ShaheerJavaid {
    public static void quickSort(int[] balances, int[] accountNumbers, int low, int high) {
        if (low < high) {
            int pivotIndex = partition(balances, accountNumbers, low, high);

            quickSort(balances, accountNumbers, low, pivotIndex - 1);
            quickSort(balances, accountNumbers, pivotIndex + 1, high);
        }
    }

    public static int partition(int[] balances, int[] accountNumbers, int low, int high) {
        int pivot = balances[high];
        int i = low - 1;

        for (int j = low; j < high; j++) {
            if (balances[j] >= pivot) {
                i++;

                int tempBalance = balances[i];
                balances[i] = balances[j];
                balances[j] = tempBalance;

                int tempAccount = accountNumbers[i];
                accountNumbers[i] = accountNumbers[j];
                accountNumbers[j] = tempAccount;
            }
        }

        int tempBalance = balances[i + 1];
        balances[i + 1] = balances[high];
        balances[high] = tempBalance;

        int tempAccount = accountNumbers[i + 1];
        accountNumbers[i + 1] = accountNumbers[high];
        accountNumbers[high] = tempAccount;

        return i + 1;
    }

    public static void printAccounts(int[] accountNumbers, int[] balances) {
        System.out.println("Account No\tBalance");
        for (int i = 0; i < accountNumbers.length; i++) {
            System.out.println(accountNumbers[i] + "\t\t" + balances[i]);
        }
    }

    public static void main(String[] args) {
        int n = 5;
        int[] accountNumbers = {1001, 1002, 1003, 1004, 1005};
        int[] balances = new int[n];

        Random random = new Random();
        for (int i = 0; i < n; i++) {
            balances[i] = random.nextInt(100001);
        }

        System.out.println("Original Account Balances:");
        printAccounts(accountNumbers, balances);
        quickSort(balances, accountNumbers, 0, n - 1);

        System.out.println("\nSorted Account Balances:");
        printAccounts(accountNumbers, balances);
    }
}
```

OUTPUT:



```
run:
Original Account Balances:
Account No      Balance
1001            41368
1002            1053
1003            64262
1004            61589
1005            64338

Sorted Account Balances:
Account No      Balance
1005            64338
1003            64262
1004            61589
1001            41368
1002            1053
BUILD SUCCESSFUL (total time: 0 seconds)
```