DESIGN ANALYSIS AND ALGORITHM

LAB 2

MERGE SORT

NAME: NAOMI GEORGE

SLOT: L25+L26+L33+L34+L13+L14

REGISTRATION NO. : 19BCE7572

COURSE CODE: CSE3004

CODE:

```
public class Merge {
void merge(int a[], int beg, int mid, int end)
{
   int i, j, k;
   int n1 = mid - beg + 1;
   int n2 = end - mid;
      int LeftArray[] = new int[n1];
      int RightArray[] = new int[n2];

   for (i = 0; i < n1; i++)
   LeftArray[i] = a[beg + i];
   for (j = 0; j < n2; j++)
   RightArray[j] = a[mid + 1 + j];

   i = 0;
   j = 0;
   k = beg;
   while (i < n1 && j < n2)
   {
      if(LeftArray[i] <= RightArray[j])
      {
         a[k] = LeftArray[i];
         i++;
      }
      else
      {
         a[k] = RightArray[j];
         j++;
      }
      k++;
```

```
    }
    while (i<n1)
    {
        a[k] = LeftArray[i];
        i++;
        k++;
    }

    while (j<n2)
    {
        a[k] = RightArray[j];
        j++;
        k++;
    }
}

void mergeSort(int a[], int beg, int end)
{
    if (beg < end)
    {
        int mid = (beg + end) / 2;
        mergeSort(a, beg, mid);
        mergeSort(a, mid + 1, end);
        merge(a, beg, mid, end);
    }
}

/* Function to print the array */
void printArray(int a[], int n)
{
    int i;
```

```
    for (i = 0; i < n; i++)
        System.out.print(a[i] + " ");
}

public static void main(String args[])
{
    int a[] = { 11, 30, 24, 7, 31, 16, 39, 41 };
    int n = a.length;
    Merge m1 = new Merge();
    System.out.println("\nBefore sorting array elements are - ");
    m1.printArray(a, n);
    m1.mergeSort(a, 0, n - 1);
    System.out.println("\nAfter sorting array elements are - ");
    m1.printArray(a, n);
    System.out.println("");
}

  }          mstSet[i] = false;
        }
        key[0] = 0;
        parent[0] = -1;
        for (int count = 0; count < V - 1; count++) {
            int u = minKey(key, mstSet);
            mstSet[u] = true;
            for (int v = 0; v < V; v++)
                if (graph[u][v] != 0 && mstSet[v] == false && graph[u][v] < key[v]) {
                    parent[v] = u;
                    key[v] = graph[u][v];
                }
        }

        printPrims(parent, graph);
```

```
    }

    public static void main(String[] args)
    {
        Prims t = new Prims();
        int graph[][] = new int[][] { { 0, 2, 0, 6, 0 },
                                      { 2, 0, 3, 8, 5 },
                                      { 0, 3, 0, 0, 7 },
                                      { 6, 8, 0, 0, 9 },
                                      { 0, 5, 7, 9, 0 } };

        t.primMST(graph);
    }
}
```

OUTPUT: