**Merge Sort with permutation and no. of merges**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

void swap(int \*a, int \*b)

{

  int temp = \*a;

  \*a = \*b;

  \*b = temp;

}

int k = 0;

void permute(int \*\*arr1, int \*arr, int start, int end)

{

  if (start == end)

  {

    for (int i = 0; i <= end; i++)

    {

      arr1[k][i] = arr[i];

    }

    k++;

  }

  else

  {

    for (int i = start; i <= end; i++)

    {

      swap(&arr[start], &arr[i]);

      permute(arr1, arr, start + 1, end);

      swap(&arr[start], &arr[i]);

    }

  }

}

void merge(int \*arr, int l, int m, int r, int \*countMerge)

{

  int i, j, k;

  int n1 = m - l + 1;

  int n2 = r - m;

  int \*L = (int \*)malloc(n1 \* sizeof(int));

  int \*R = (int \*)malloc(n2 \* sizeof(int));

  for (i = 0; i < n1; i++)

    L[i] = arr[l + i];

  for (j = 0; j < n2; j++)

    R[j] = arr[m + 1 + j];

  i = 0;

  j = 0;

  k = l;

  while (i < n1 && j < n2)

  {

    if (L[i] <= R[j])

    {

      arr[k] = L[i];

      i++;

    }

    else

    {

      arr[k] = R[j];

      j++;

    }

    k++;

    (\*countMerge)++;

  }

  while (i < n1)

  {

    arr[k] = L[i];

    i++;

    k++;

  }

  while (j < n2)

  {

    arr[k] = R[j];

    j++;

    k++;

  }

}

void mergeSort(int \*arr, int l, int r, int \*countMerge)

{

  if (l < r)

  {

    int m = (l + r) / 2;

    mergeSort(arr, l, m, countMerge);

    mergeSort(arr, m + 1, r, countMerge);

    merge(arr, l, m, r, countMerge);

  }

}

int main()

{

  srand(time(NULL));

  int \*\*arr, \*a;

  FILE \*f;

  int n, i, j, k;

  printf("Enter the number of elements: ");

  scanf("%d", &n);

  a = (int \*)malloc(n \* sizeof(int));

  for (i = 0; i < n; i++)

  {

    int num = rand() % 100;

    a[i] = num;

  }

  f = fopen("output.txt", "w");

  if (f == NULL)

  {

    printf("Error opening file\n");

    exit(1);

  }

  fprintf(f, "The array is: ");

  for (i = 0; i < n; i++)

  {

    fprintf(f, "%d ", a[i]);

  }

  fprintf(f, "\n\n");

  int fact = 1;

  for (i = 1; i <= n; i++)

  {

    fact \*= i;

  }

  arr = (int \*\*)malloc(fact \* sizeof(int \*));

  int \*countMerge = (int \*)malloc(fact \* sizeof(int));

  for (i = 0; i < fact; i++)

  {

    arr[i] = (int \*)malloc(n \* sizeof(int));

  }

  permute(arr, a, 0, n - 1);

  for (i = 0; i < fact; i++)

  {

    fprintf(f, "Permutation: ");

    for (j = 0; j < n; j++)

    {

      fprintf(f, "%d ", arr[i][j]);

    }

    fprintf(f, "\n");

    fprintf(f, "Sorted permutation: ");

    countMerge[i] = 0;

    mergeSort(arr[i], 0, n - 1, &countMerge[i]);

    for (j = 0; j < n; j++)

    {

      fprintf(f, "%d ", arr[i][j]);

    }

    fprintf(f, "\n");

    fprintf(f, "Number of times comparison is done: %d\n\n", countMerge[i]);

  }

  printf("Results stored in 'output.txt'.\n");

  fclose(f);

  return 0;

}