Day 2 Assignment

a.) Wap to implement merge sort algorithm on a randomly generated array that is stored in input.txt and give output in the output.txt in c language

Code-

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define SIZE 10


void merge(int arr[], int left, int mid, int right) {
    int i, j, k;
    int n1 = mid - left + 1;
    int n2 = right - mid;
    int L[n1], R[n2];

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

    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++;
    }
}

// Merge Sort function
void mergeSort(int arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr, left, mid, right);
    }
}

int main() {
    FILE *inputFile = fopen("input.txt", "w");
    if (inputFile == NULL) {
        printf("Error opening input file for writing!\n");
        return 1;
    }

    srand(time(0));
    for (int i = 0; i < SIZE; i++) {
        fprintf(inputFile, "%d ", (rand() % SIZE));
    }
    fclose(inputFile);

    inputFile = fopen("input.txt", "r");
    if (inputFile == NULL) {
        printf("Error opening input file for reading!\n");
        return 1;
    }

    int arr[SIZE], n = 0;
    while (fscanf(inputFile, "%d", &arr[n]) != EOF) {
        n++;
    }
    fclose(inputFile);

    mergeSort(arr, 0, n - 1);

    FILE *outputFile = fopen("output.txt", "w");
    if (outputFile == NULL) {
        printf("Error opening output file!\n");
        return 1;
    }

    for (int i = 0; i < n; i++) {
        fprintf(outputFile, "%d ", arr[i]);
    }
    fclose(outputFile);

    printf("Sorting complete. Check output.txt\n");
    return 0;
}
```

Input.txt

```
5 6 0 6 8 9 2 8 4 0
```

Output.txt

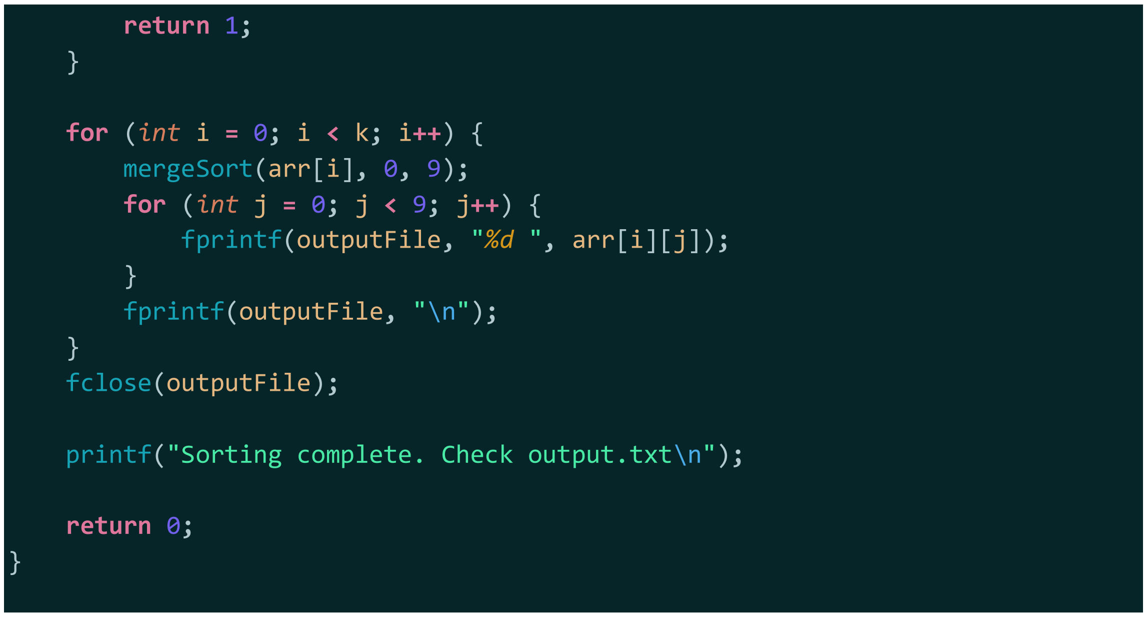
```
0 0 2 4 5 6 6 8 8 9
```

b.) Implement the same merge sort on 'k' different number of arrays where k will be the input from the user

Code-

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define SIZE 100


void merge(int arr[], int left, int mid, int right) {
    int i, j, k;
    int n1 = mid - left + 1;
    int n2 = right - mid;
    int L[n1], R[n2];

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

    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++;
        }
```

```c
        k++;
    }

    while (i < n1) {
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < n2) {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergeSort(int arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr, left, mid, right);
    }
}

int main() {
    int k;
    printf("Enter the number of arrays (k): ");
    scanf("%d", &k);

    srand(time(0));

    FILE *inputFile = fopen("input.txt", "w");
    if (inputFile == NULL) {
        printf("Error opening input.txt for writing!\n");
        return 1;
    }

    int arr[k][10];

    for (int i = 0; i < k; i++) {
        for (int j = 0; j < 10; j++) {
            arr[i][j] = (rand() % SIZE);
            fprintf(inputFile, "%d ", arr[i][j]);
        }
        fprintf(inputFile, "\n");
    }
    fclose(inputFile);

    FILE *outputFile = fopen("output.txt", "w");
    if (outputFile == NULL) {
        printf("Error opening output.txt for writing!\n");
```



```c
        return 1;
    }

    for (int i = 0; i < k; i++) {
        mergeSort(arr[i], 0, 9);
        for (int j = 0; j < 9; j++) {
            fprintf(outputFile, "%d ", arr[i][j]);
        }
        fprintf(outputFile, "\n");
    }
    fclose(outputFile);

    printf("Sorting complete. Check output.txt\n");

    return 0;
}
```

Input.txt

```
66 49 51 7 82 18 51 46 42 82
94 32 74 16 29 48 63 13 86 10
10 69 28 65 90 39 14 20 99 49
39 35 67 76 29 71 54 30 40 51
5 93 24 8 72 99 6 32 40 98
10 14 14 54 60 45 69 52 57 1
34 78 52 14 6 56 86 72 75 61
97 67 39 23 76 12 14 91 8 24
14 52 10 63 27 13 65 7 48 61
72 83 0 7 3 66 2 54 90 72
```

Output.txt

```
7 18 42 46 49 51 51 66 82
10 13 16 29 32 48 63 74 86
10 14 20 28 39 49 65 69 90
29 30 35 39 40 51 54 67 71
5 6 8 24 32 40 72 93 98
1 10 14 14 45 52 54 57 60
6 14 34 52 56 61 72 75 78
8 12 14 23 24 39 67 76 91
7 10 13 14 27 48 52 61 63
0 2 3 7 54 66 72 72 83
```