

Aayush Shah

19BCE245

17 August 2021

Design and Analysis of Algorithms

Practical 3

• Code :

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <sys/stat.h>

int finalArr[500];

int getRandomNumber(int lower, int upper){
    return ((rand() % (upper - lower + 1)) + lower);
}

void swap(int* a, int* b){
    int t = *a;
    *a = *b;
    *b = t;
}

int partition (int arr[], int low, int high)
{
    int pivot = arr[high];
    int i = (low - 1);
    for (int j = low; j <= high - 1; j++){
        if (arr[j] < pivot){
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return (i + 1);
}
```

```
}

void quickSort(int arr[], int low, int high){
    if (low < high){
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

void printArray(int A[], int size){
    int i;
    for (i = 0; i < size; i++)
        printf("%d ", A[i]);
    printf("\n");
}

void merge(int arr1[], int arr2[], int n1, int n2start, int
n2end){

    int i, j, k=0;
    int n2 = 50;
    int arr3[500];

    int L[n1], R[50];

    for (i = 0; i < n1; i++)
        L[i] = arr1[i];
    for (j = n2start; j < n2+n2start; j++)
        R[k++] = arr2[j];

    i = 0;
    j = 0;
    k = 0;
    while (i < n1 && j < 50) {
        if (L[i] <= R[j]) {
            arr3[k] = L[i];
            i++;
        }
        else {
            arr3[k] = R[j];
            j++;
        }
    }
}
```

```
        k++;
    }

    while (i < n1) {
        arr3[k] = L[i];
        i++;
        k++;
    }

    while (j < 50) {
        arr3[k] = R[j];
        j++;
        k++;
    }

    char fileName[100];
    sprintf(fileName, "./sorted/%d.txt", n2end);
    printf("%d Numbers merged and sorted at %s.\n", n2end, fileName);
    FILE* filePointer = fopen(fileName, "w");

    char str[40];
    for (int z=0; z<k; z++) {
        finalArr[z] = arr3[z];
        sprintf(str, "%d\n", arr3[z]);
        fputs(str, filePointer);
    }
    fclose(filePointer);
}

int main(){
    int lower = 1, upper = 10000, count = 500;
    srand(time(0));
    int tempInt;
    char str[12];
    int extractedNumbers[count];
    char *line = NULL;
    size_t len = 0;
    ssize_t read;

    FILE *filePointer ;
    filePointer = fopen("random_numbers.txt", "w");
```

```
// Generating random numbers and adding in file
for (int i=0; i<count; i++) {
    tempInt = getRandomNumber(lower, upper);
    sprintf(str, "%d\n", tempInt);
    fputs(str,filePointer);
}

fseek(filePointer, 0, SEEK_SET);

//Extracting the numbers
filePointer = fopen("random_numbers.txt", "r");

if(filePointer==NULL){
    printf("File doesn't exists.");
}
else {
    int counter = 0;
    while ((read = getline(&line, &len, filePointer)) != -1) {
        extractedNumbers[counter++] = atoi(line);
    }
}
fclose(filePointer);

mkdir("sorted",0777); //created directory named sorted

int size;
int counter = 0;

for (int i=0; i<count/50; i++) {

    quickSort(extractedNumbers, i*50, (i+1)*50-1);

    merge(finalArr, extractedNumbers, counter, i*50,
(i+1)*50);

    counter+=50;
}

fclose(filePointer);

return 0;
}
```

• Output

```

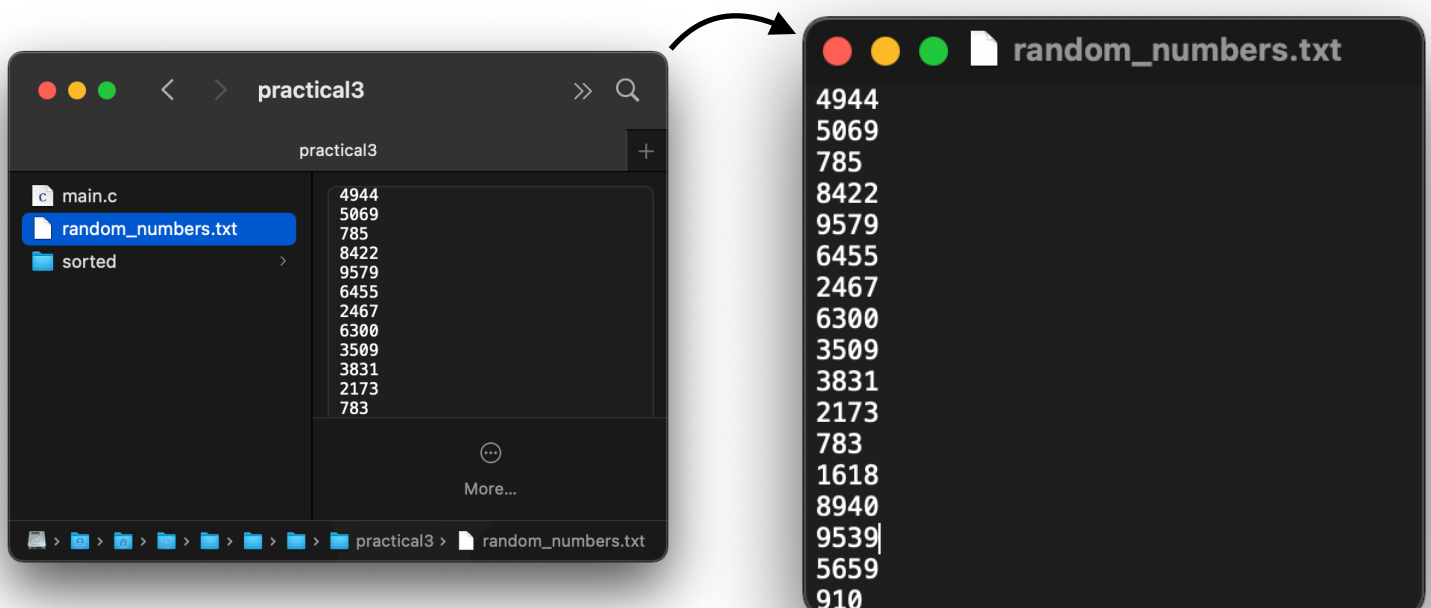
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #include <string.h>
5  #include <sys/stat.h>
6
7  int finalArr[500];
8
9  int getRandomNumber(int lower, int upper){
10     return ((rand() % (upper - lower + 1)) + lower);
11 }
12
13 void swap(int* a, int* b){
14     int t = *a;
15     *a = *b;
16     *b = t;
17 }
18
19 int partition (int arr[], int low, int high){
20 {
21     int pivot = arr[high];

```

50 Numbers merged and sorted at ./sorted/50.txt.
100 Numbers merged and sorted at ./sorted/100.txt.
150 Numbers merged and sorted at ./sorted/150.txt.
200 Numbers merged and sorted at ./sorted/200.txt.
250 Numbers merged and sorted at ./sorted/250.txt.
300 Numbers merged and sorted at ./sorted/300.txt.
350 Numbers merged and sorted at ./sorted/350.txt.
400 Numbers merged and sorted at ./sorted/400.txt.
450 Numbers merged and sorted at ./sorted/450.txt.
500 Numbers merged and sorted at ./sorted/500.txt.

Run Succeeded | Time 52 ms | Peak Memory 766K | Symbol | Spaces: 2 | Line 7, Column 19

Random numbers generated file :



Sorted numbers' individual files generated in 'sorted' folder :

