

CSE331: Data Structures and Algorithms

Heap Sort Lab Report



Name: Anthony Amgad Fayek

Program: CESS

ID: 19P9880

**The Full Project is in a
GitHub Repository
Below**

Here are the used libraries and definitions:

```
#include <iostream>
#include <fstream>
#include <ctime>
#include <cstdlib>
#define LENGTH 10000
using namespace std;
```

Part 1:

Writing a C++ function to generate 10,000 random numbers between 1 and 10,000 and save them in a file (the full generated txt is in the GitHub repository linked below):

```
void createRandFile() {
    ofstream mfile("unsortedFile.txt");
    srand(time(0));

    for (int i = 0; i < LENGTH; i++) {
        mfile << ((rand() % LENGTH) + 1) << endl;
    }
}
```

Part 2:

Writing the Heap sort functions (this includes the counter (the variable “step”) that is required in Part 3:

```
int maxHeapify(int arr[], int i, int len) {
    int step = 3;
    int l = (2 * i) + 1;
    int r = (2 * i) + 2;
    int largest;
    if (l < len && arr[l] > arr[i]) {
        largest = l;
        step++;
    }
    else {
        largest = i;
        step++;
    }
    if (r < len && arr[r] > arr[largest]) {
        largest = r;
        step++;
    }
    if (largest != i) {
        int temp = arr[i];
        arr[i] = arr[largest];
        arr[largest] = temp;
        step += 3;
        step += maxHeapify(arr, largest, len);
    }
    return step;
}
```

```

int buildMaxHeap(int arr[], int l) {
    int step = 0;
    for (int i = l / 2; i >= 0; i--) {
        maxHeapify(arr, i, l);
        step++;
    }
    return step;
}

int heapSort(int arr[], int l) {
    int temp, step=0;
    step += buildMaxHeap(arr, l);
    for (int i = l-1; i > 0; i--) {
        temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;
        step += 3;
        step += maxHeapify(arr, 0, i);
    }
    return step;
}

```

Writing a function to write the resultant array into a file (the full generated txt is in the GitHub repository linked below):

```

void createSortedFile(int arr[]) {
    ofstream mfile("sortedFile.txt");
    for (int i = 0; i < LENGTH; i++) {
        mfile << arr[i] << endl;
    }
}

```

Part 3:

Creating the main function which reads n items using another function from the file generated and executes the insertion algorithm with step 50 and writes a file that includes pairs of n and f(n) ("step") (the full generated txt is in the GitHub repository linked below):

```

void readFile(int arr[], int l) {
    ifstream mfile("unsortedFile.txt");
    for (int i = 0; i < l; i++) {
        mfile >> arr[i];
    }
}

int main() {
    int arr[LENGTH];
    createRandFile();
    readFile(arr, LENGTH);
    ofstream sFile("stepFile.txt");
    int x[LENGTH];
    for (int i = 10; i < LENGTH; i += 50) {
        for (int j = 0; j < i; j++) {
            x[j] = arr[j];
        }
        sFile << i << ',' << heapSort(x, i) << endl;
    }
}

```

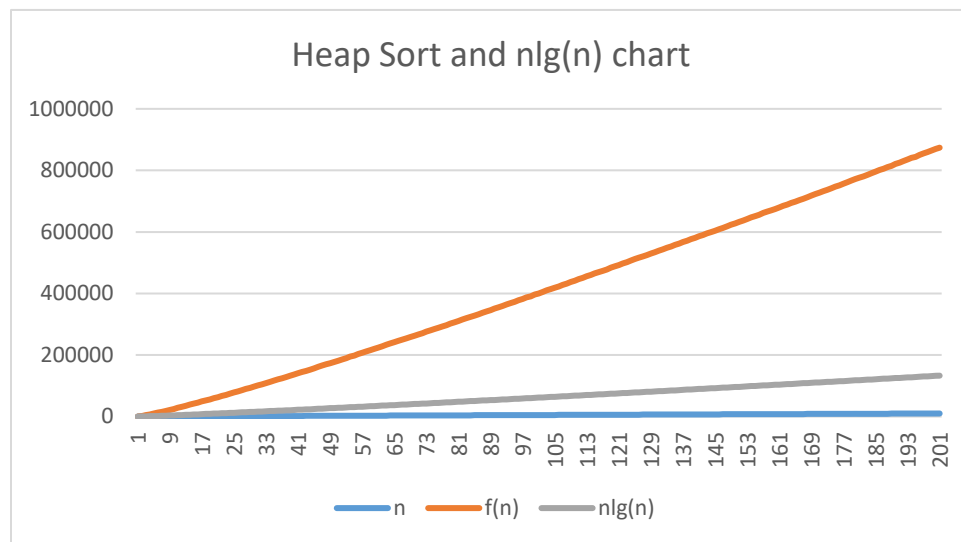
```

}
sFile << LENGTH << ',' << heapSort(arr, LENGTH) << endl;
createSortedFile(arr);
system("pause");
return 0;
}

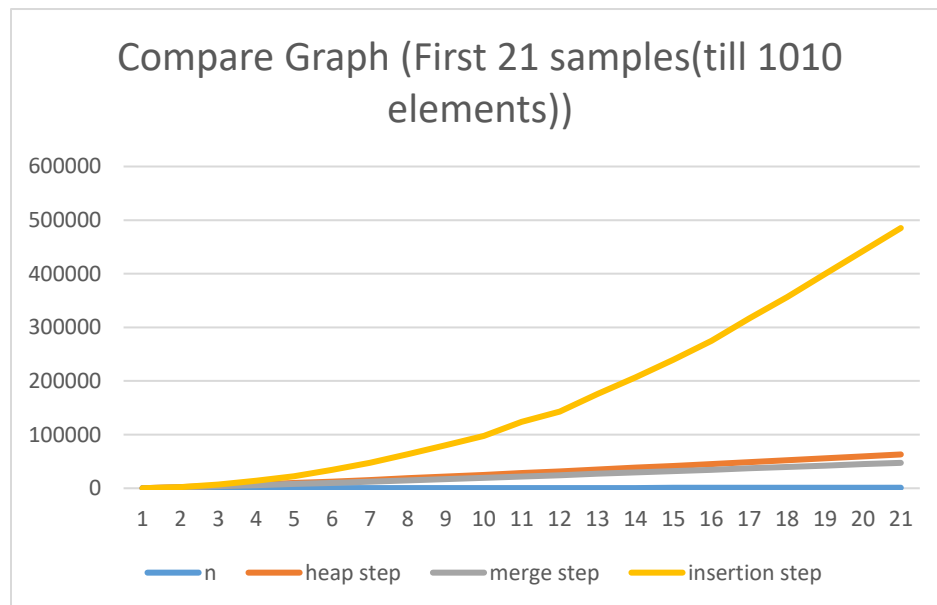
```

Part 4:

The “stepFile.txt” created in the main function is then imported into excel with an added column of $(n \lg(n))$. Then a generated Graph from the excel is created:



Another graph that compares merge, insertion and heap sort is created:



GitHub Repository:

<https://github.com/Anthony-Amgad/CSE331HeapSort19P9880>