# *CSE331: Data Structures and Algorithms*

***Insertion 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 insertion sort function (this includes the counter (the variable “step”) that is required in Part 3:

int insertionSort(int arr[], int len) {

int step = 0;

for (int i = 1; i < len; i++) {

int key = arr[i];

step++;

int j = i - 1;

step++;

while (j >= 0 && arr[j] > key) {

arr[j + 1] = arr[j];

step++;

j--;

step++;

}

arr[j + 1] = key;

step++;

}

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;

}

}

Additional Part:

Writing a testing function to check if the file has been correctly sorted:

bool testSortedFile() {

int arr[LENGTH];

ifstream mfile("sortedFile.txt");

for (int i = 0; i < LENGTH; i++) {

mfile >> arr[i];

}

for (int i = 0; i < LENGTH - 1; i++) {

if (arr[i] > arr[i + 1])

return false;

}

return true;

}

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 < 10000; i += 50) {

for (int j = 0; j < i; j++) {

x[j] = arr[j];

}

sFile << i << ',' << insertionSort(x, i) << endl;

}

sFile << 10000 << ',' << insertionSort(arr, 10000) << endl;

createSortedFile(arr);

if(testSortedFile())

cout << "Test for sort is succesful" << endl;

else

cout << "Test for sort is not succesful" << endl;

system("pause");

return 0;

}

Part 4:

The “stepFile.txt” created in the main function is then imported into excel with an added column of (n2). Here’s a sample of the table in excel (the full generated excel is in the GitHub repository linked below):

|  |  |  |
| --- | --- | --- |
| n | step | n2 |
| 10 | 51 | 100 |
| 60 | 1971 | 3600 |
| 110 | 6879 | 12100 |
| 160 | 14389 | 25600 |
| 210 | 22247 | 44100 |
| 260 | 34269 | 67600 |
| 310 | 47711 | 96100 |
| 360 | 63457 | 129600 |
| 410 | 80219 | 168100 |
| 460 | 97435 | 211600 |

Then a generated Graph from the excel is created:

GitHub Repository:

<https://github.com/Anthony-Amgad/CSE331InsertionSort19P9880>