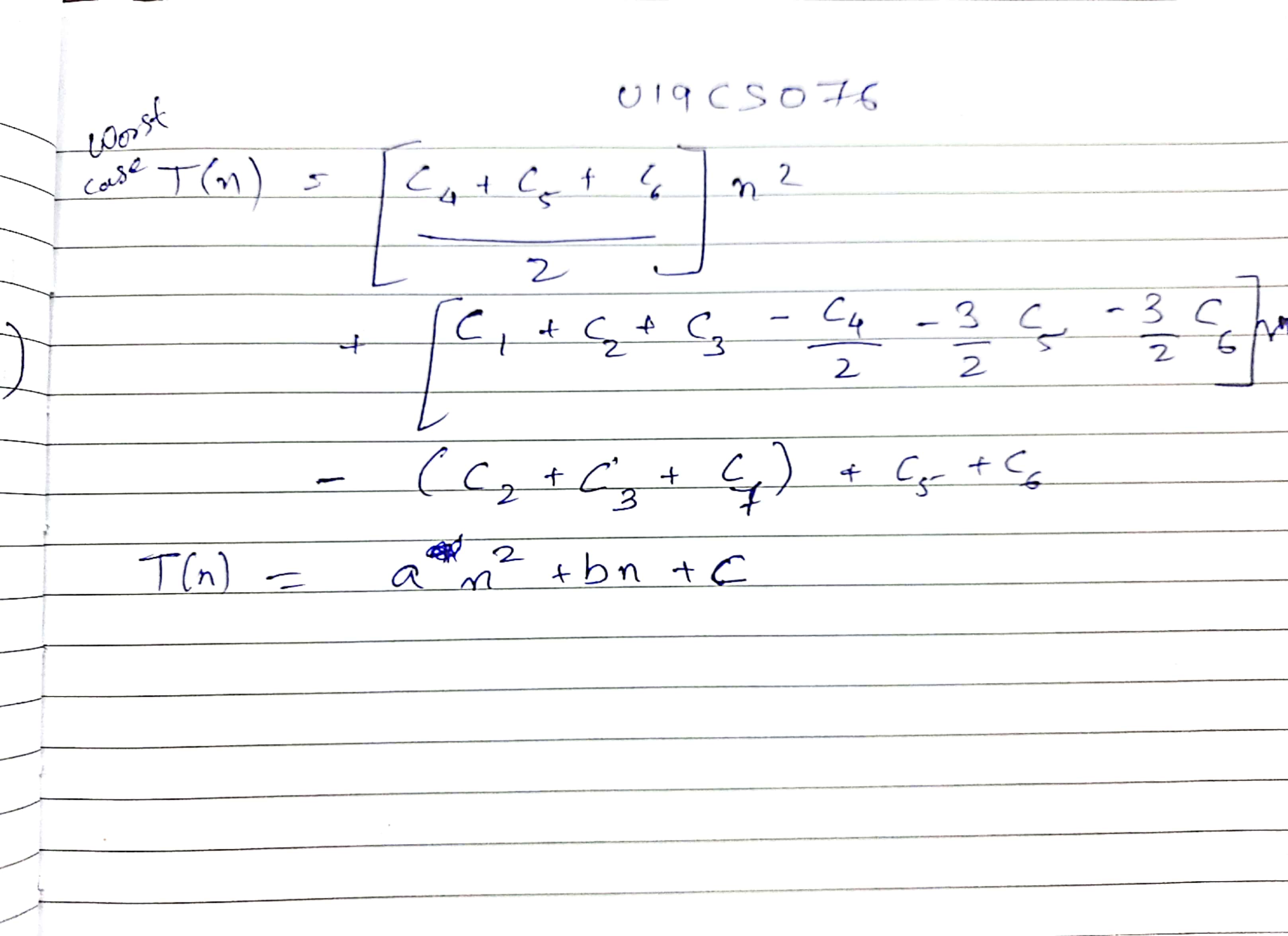
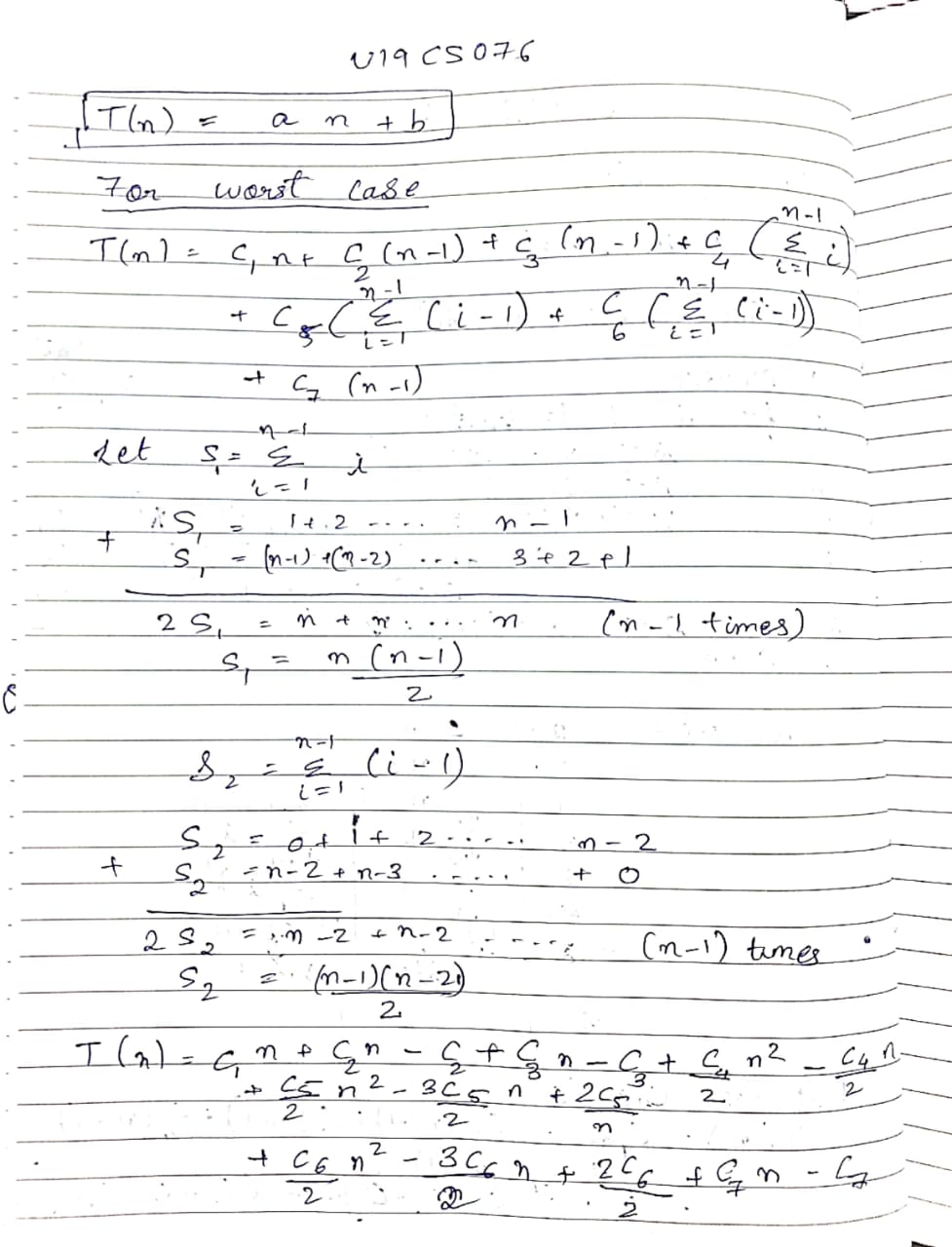
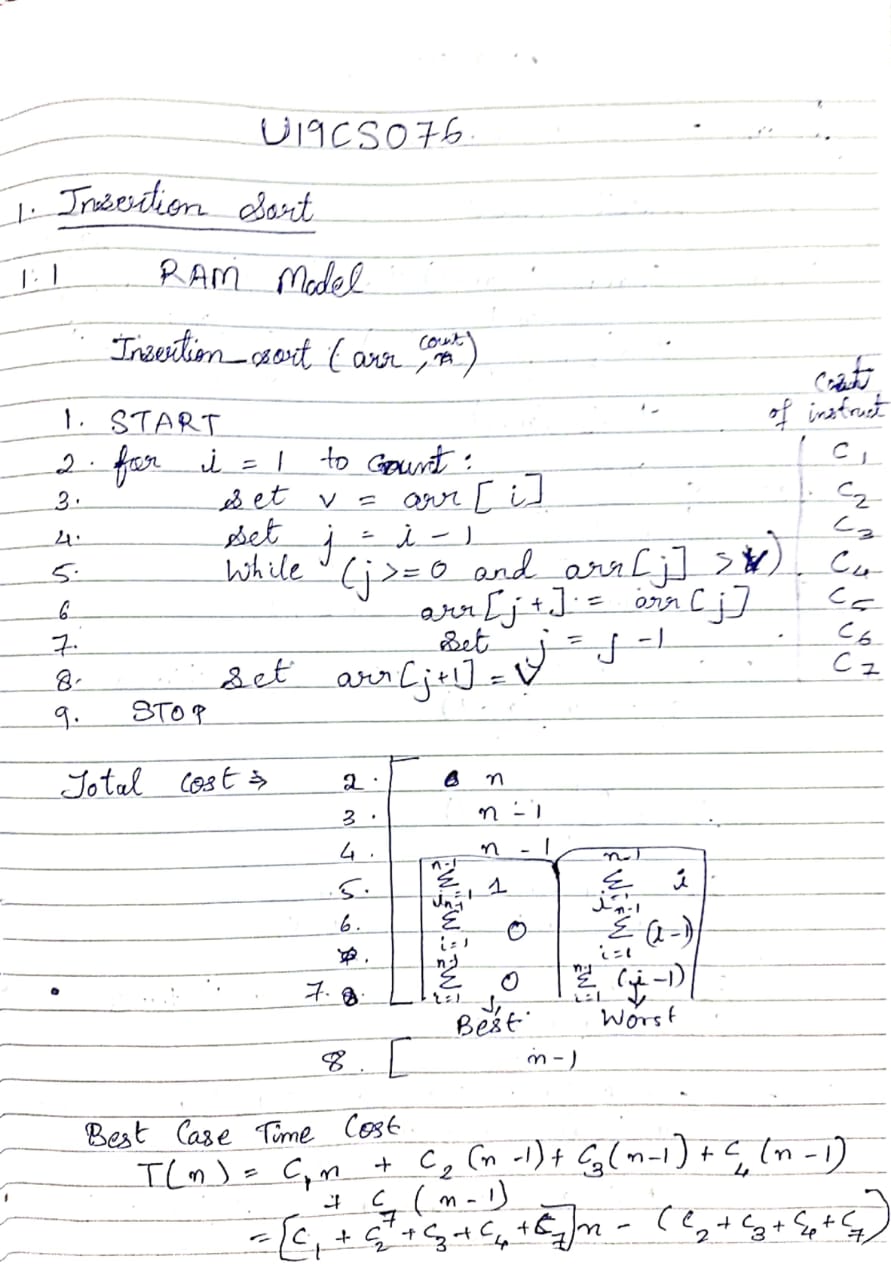
U19CS076 DAA ASSIGNMENT 2

KRITHIKHA BALAMURUGAN

• Insertion sort

Q1. Analyze the time complexity of the above algorithm using the RAM model

Q2. Implement the above algorithm using the programming language of your

choice.

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<time.h>

clock\_t begin, end;

double time\_;

void insertion\_sort(long long data[], long long count)

{

long long i, v, j;

for (i = 1; i < count; i++)

{

v = data[i];

j = i-1;

while (j>=0 && data[j]>v)

{

data[j+1] = data[j];

j = j-1;

}

data[j+1] = v;

}

}

long long count(char file[])

{

FILE \*fp = fopen(file, "r");

long long count = 0;

char b[100];

while(fscanf(fp, "%s\n", &b) == 1)

count++;

fclose(fp);

return count;

}

void insertion\_sort\_desc(long long data[], long long count)

{

long long i, v, j;

for (i = 1; i < count; i++)

{

v = data[i];

j = i-1;

while (j>=0 && data[j]<v)

{

data[j+1] = data[j];

j = j-1;

}

data[j+1] = v;

}

}

int main()

{

long long j;

long long n;

long long \*data; //array to hold data

int i;

char filename[15];

FILE \*fp;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TIME SUMMARY\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for(i=0;i<10;i++)

{

sprintf(filename, "File %d.txt", i+1);

n = count(filename);

printf("-------------------------File %d.txt-----------------------\n",i+1);

printf("File %d has %lld elements\n",i+1,n);

fp = fopen(filename, "r");

data=(long long\*)malloc(n\*((long long)sizeof(long long)));

for(j=0; j<n; j++)

{

fscanf(fp, "%lld", &data[j]);

}

begin= clock();

insertion\_sort(data, n);

end = clock();

fclose(fp);

time\_ = ((double)(end-begin)) / CLOCKS\_PER\_SEC;

printf("Average case : %0.10lf\n", time\_);

sprintf(filename, "File %d\_asc.txt", i+1);

fp = fopen(filename, "r");

for(j=0; j<n; j++)

{

fscanf(fp, "%lld", &data[j]);

}

begin = clock();

insertion\_sort(data, n);

end = clock();

fclose(fp);

time\_= ((double)(end-begin)) / CLOCKS\_PER\_SEC;

printf("Best case : %0.10lf\n",time\_);

fp = fopen(filename, "r");

for(j=n-1; j>=0; j--)

{

fscanf(fp, "%lld", &data[j]);

}

begin = clock();

insertion\_sort\_desc(data, n);

end = clock();

fclose(fp);

time\_ = ((double)(end-begin)) / CLOCKS\_PER\_SEC;

printf("Worst case %0.10lf\n\n", time\_);

free(data);

}

}

Q3.Provide the details of Hardware/Software you used to implement the algorithm and to measure the time.

Compiler Dev C++ 5.11

OS Name Microsoft Windows 10 Home (i5 8th Gen)

Version 10.0.19042 Build 19042

System Name DESKTOP-BLE6CMQ

System Model HP Pavilion x360 Convertible 14-ba1xx

System Type x64-based PC

Processor Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz, 1800 Mhz, 4 Core(s), 8 Logical Processor(s)

BIOS Version/Date Insyde F.54, 04-12-2019

Installed Physical Memory (RAM) 8.00 GB

Total Physical Memory 7.88 GB

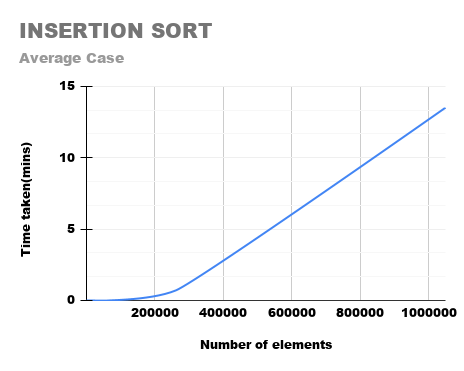
Available Physical Memory 1.75 GB

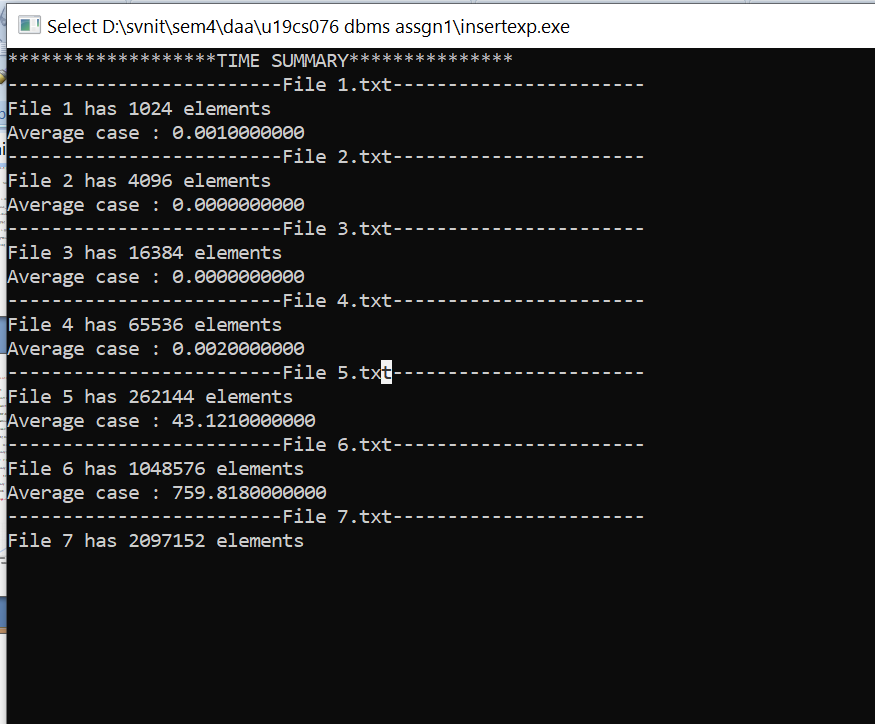
Total Virtual Memory 12.4 GB

Available Virtual Memory 4.59 GB

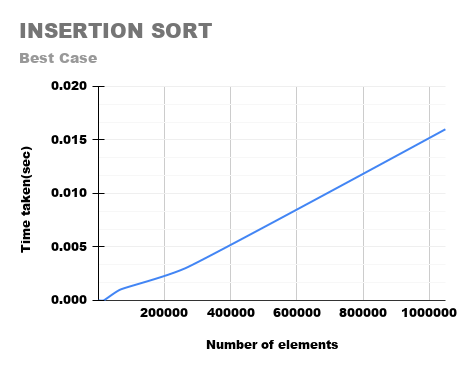
Page File Space 4.50 GB

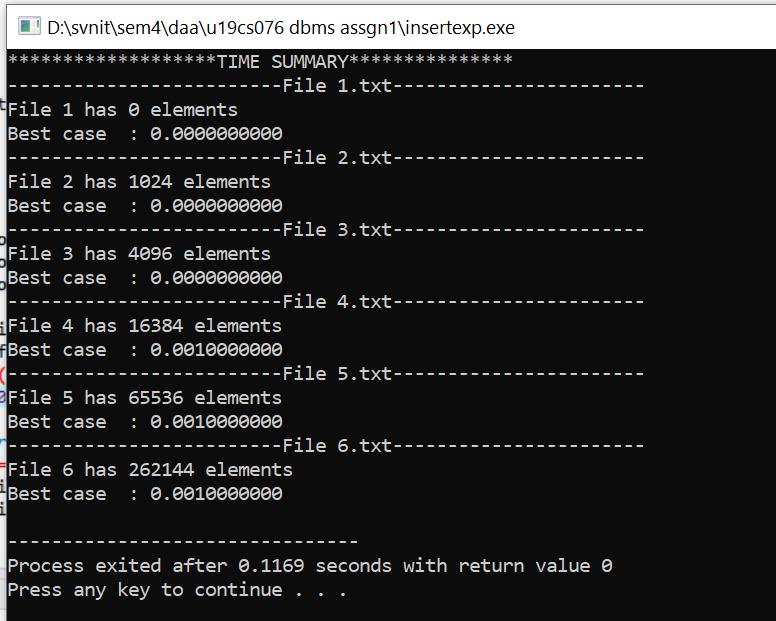
Q5. Measure the average-case time (considering current data of ten files) of insertion sort for all ten files. Plot a graph.



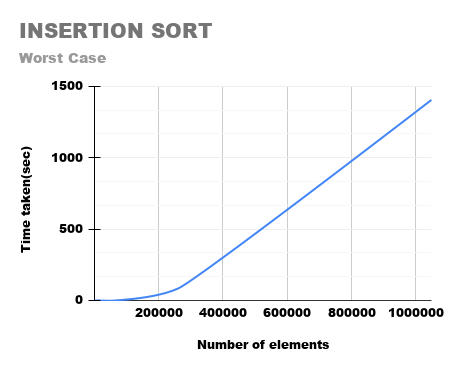


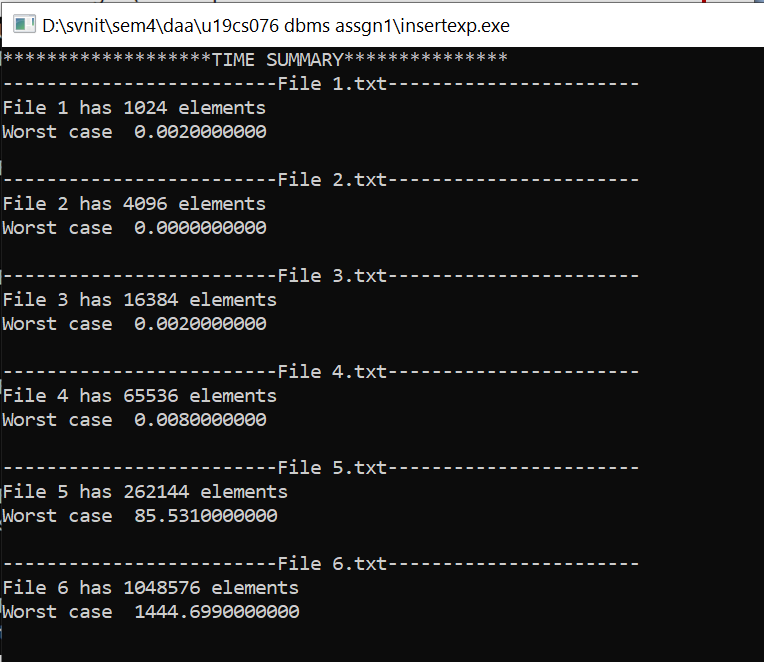
Q6. Measure the best-case time of insertion sort for all ten files. Plot a graph.



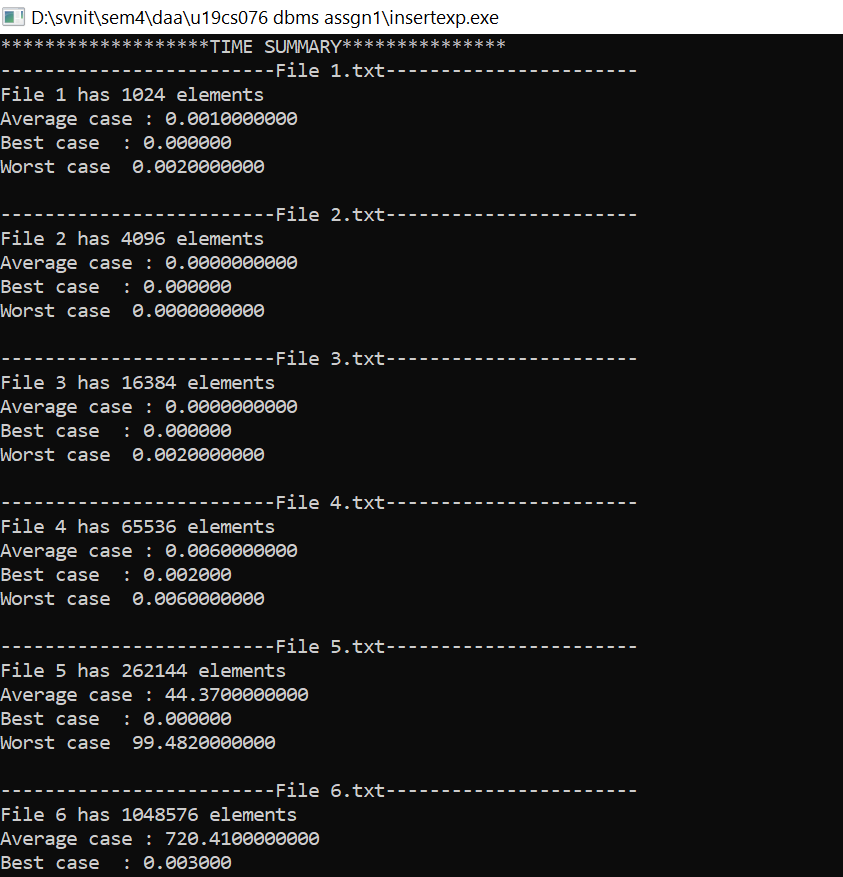


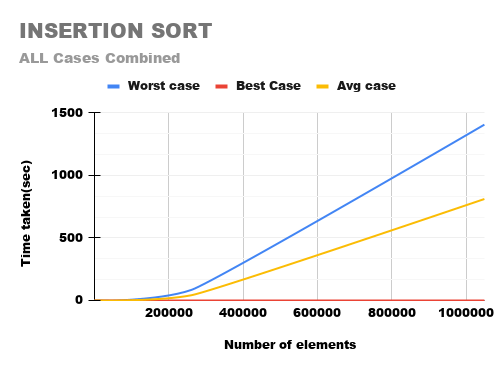
Q7. Measure the worst-case time of insertion sort for all ten files. Plot a graph.





**ALL CASES COMBINED OUTPUT**





Q8. Assume that you don’t know the time complexity of above algorithms.

8.1. Can you predict the same based on your implementation?

By observing Graphs of 3 cases we can conclude:

Best case is a Straight Line -> An+B

Average Case is a Parabolic graph -> An^2+Bn+C

Worst Case is a Parabolic graph -> An^2+Bn+C

So we can predict values if we know the input N by using extrapolating graphs.

8.2. Do they match with theoretical time complexity? Yes/No.

Yes, they would match with theoretical values too.

8.3. If yes, then write the time complexity of each algorithm. If no, then write

the difference.

According to observations and graphs given above:

Time Complexity of Best Case-> An+B -> BIG THETA(n)

Time Complexity of Worst Case-> An2+Bn+C -> BIG THETA(n2)