数据结构

课程设计(论文)

题目： Employee System Management

院（系）：**Computer Science 学院**

专业班级：**computer**

学　号：**1**0**0**0****

学生姓名：**D. Ell Bouss**

指导教师：********

教师职称：副教授

起止时间：**2019.1.7 至 2019.1.11**

课程设计（论文）任务及评语

院（系）：电子与信息工程学院　　教研室：软件工程

学 号	1**0**0**	学生姓名	D. Ell Bouss	专业班级	2017 computer
课程设计（论文）题目	**Employee Management System**				
课程设计（论文）任务	1.content （1）Create Employees Database (records: Name, ID, Designation, Age, Salary, Years of Experience, Gender). （2）Output the list of employees with all their records; （3）Search one specific employee’s records; （4）Modify Employees records; （5）Insert new employees records; （6）Remove employees records; 2.demand when writing program： （1）Define the storage structure yourself; （2）To each demand, define a sub function, call the sub function in main function. （3）Compile and run the program ,record the run result and analyze the result;				
指导教师评语及成绩	平时成绩：_____ 答辩成绩：____ 论文成绩： 总成绩：_____ 指导教师签字： 年　月　日				

注：平时成绩占 20%，答辩成绩占 40%，论文成绩占 40

1. FUNCTION OF SUBJECT

The selected subject is: **Employee Management System**.

Process:

Coding a software (program) using C++ to store employees' records (name, id, gender, salary, …) in the most efficient way I can, and allowing users to do some basic operations in Data Structure, for example: Search, Insert, Modify, Delete, …
I have named said program
" **Data Management Algorithms by D…** " (**DMA**),
from the beginning until the end of the course design I have looked for ways to systematically improve the program to its most efficient capability: running Beta Test of the program over and over again, going from one IDE to another (Microsoft Visual Studio, CodeBlocks, Dev C++…), the point is to minimize the bugs that occur when the program is running (indeed Building, Compiling, Debugging, and Running the code source or project will result no errors) thus although small bugs occur in the program, they are on the most minimal scale and also, they are inexistent when carefully following the instructions.
This program allows the user to implement 6 of the most useful operations in Data Structure on a console application:

- Initialize or Create a list.
- Output the list.
- Search for records or data in the list.
- Modify records or data in the list.
- Insert new records or data in the list.
- Delete records or data from the list.

2. DATA STRUCTURE

The chosen structure for the program is: **Linear Structure**

Advantages of linear structure:

1) Only one and the last data element.
2) Only one and the first data element.
3) Except the last, each data element has one next element.
4) Except the first, each data element has one prior element.

Linear programming is most suitable for solving complex problems. Helps in simplicity and productive management of an organization which gives better outcomes.

Improves quality of decision: A better quality can be obtained with the system by making use of linear programming.

Provides a way to unify results from disparate areas of mechanism design.

More flexible than any other system, a wide range of problems can be solved easily, saves memory, space and also provides faster access to data.

For a better optimization of the **DMA** a file “data.txt” is included to the process, to save all data, permanently (even after shutting down the program).

3. PHYSICAL STRUCTURE

The physical structure chosen is: **Sequential Storage Structure (sequential access file)**.

Sequential access memory (SAM) is a class of data storage devices that read stored data in a sequence. This is in contrast to random access memory (RAM) where data can be accessed in any order. Sequential access devices are usually a form of magnetic storage or optical storage. The program uses a file “data.txt” to save the data relative to the program.

Advantages of sequential access file:

- A computer program makes a sequential file simply by writing data records, one after the other, into a newly created file area. The records may all have the same length, or lengths may vary. If they vary, each record ends with a special character or characters; when a program subsequently reads the file, it uses the characters to determine where one record ends and the next begins. Programs read sequential files the same way they were created: beginning with the first record and continuing, in ascending order, to the end.

- Compared to direct-access files, programs process sequential access files faster. Programs read direct-access file records in any order, but that flexibility comes at the price of slower performance. The positioning mechanism of the computer's hard drive works much less for sequential files than direct files, and the computer's central processing unit (CPU) likewise has less work with a sequential file.

- Sequential files are easy to read because of their simple organization. It is a simple matter to write new programs to read existing sequential files, since the program reads the records as a simple series until it encounters an end-of-file (EOF) mark.

- It is simple to program and easy to design.
- Sequential file is best use for storage space.
- In direct access file, sorting of the records are not required.
- It accesses the desired records immediately.
- It updates several files quickly.
- It has better control over record allocation.

Array structure: each data element

Name	[1] , [2] , [3] ,…, [n]
ID	[1] , [2] , [3] ,…, [n]
Designation	[1] , [2] , [3] ,…, [n]
Age	[1] , [2] , [3] ,…, [n]
Salary	[1] , [2] , [3] ,…, [n]
Years of Experience	[1] , [2] , [3] ,…, [n]
Gender	[1] , [2] , [3] ,…, [n]

4. PROGRAMMING CODE

(1) List of header files

Total number of Header Files: 16

```
#include <windows.h>
#include <stdio.h>                //contains printf, scanf …
#include <conio.h>                //contains delay(), getch(), gotoxy(), ...
#include <stdlib.h>
#include <string.h>               //contains strcmp(), strcpy(), strlen(), …

#include <iostream>
#include <fstream>                //enable File (.txt) options
#include <string>
#include <iomanip>
#include <cstdlib>
#include <algorithm>
#include <deque>
#include <functional>
#include <ctime>
#include <cstring>
#include <sstream>
#include <cstdio>
#include <cctype>
using namespace std;
```

(2) List of Variables

Total number of Variables: 35

Global Scope

```
int seconds = 5;
```

Class Employee's Scope — Variables for employee details & Interaction with User

```
char name[99][35];          //To save the Names of the employees
char id[99][8];             //To save the ID of the employees
char designation[99][25];   //To save the Designation of the employees
char age[99][3];            //To save the Ages of the employees
char salary[99][12];        //To save the Salary of the employees
char experience[99][3];     //To save the Years of Experience of the employees
char gender[99][2];         //To save the Gender of the employees
char checkName[35];
char checkId[8];
char checkDesignation[25];
char checkAge[3];
char checkSalary[12];
char checkExp[3];
int n;                      //The size of the list of employees
int i;
int userChoice;
int isRemoved;
int isRenamed;
```

Main Function’s Scope

employee e;

(Other Variables declared inside Sub-Functions)

(3) List of Functions

Total number of Functions: 21

// Utility Functions

void waitForEnter(void)

void options(void)

//Login Process and Main Menu

int login()

void fail()

//When enter a wrong password

void timerFail()

//Data Operation Functions

void initial(void) //1st Function to Create records

void output(void) //2nd Function to Output the data

void search(void) //3rd Function to Search an employee's data

void modifyRecords(void) //4th Function to Modify records (Main Function amongst Modify Functions)

void insert(void) //5th Function to Insert a new employee's records

void deleteRecords(void) //6th Function to Remove records (Main Function amongst Delete Functions)

//Main Function

int main ()

(4) List of Class

Total number of Class: 1

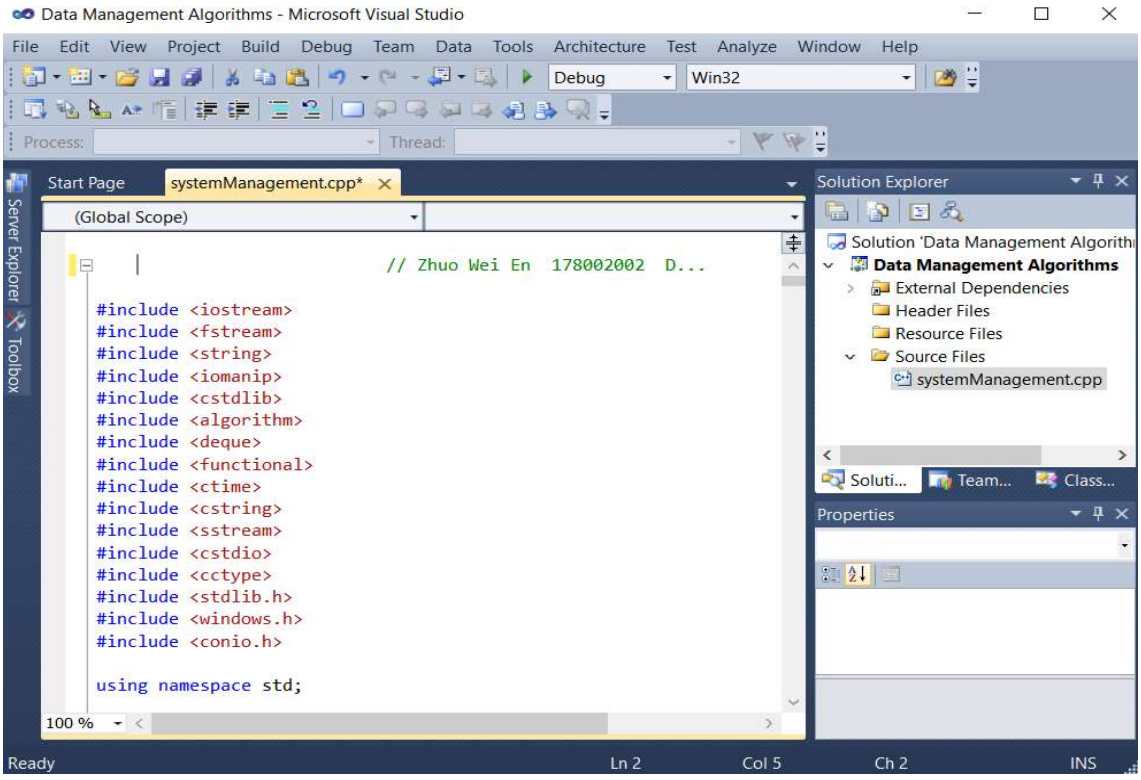
class employee //Class with two scopes

Private:

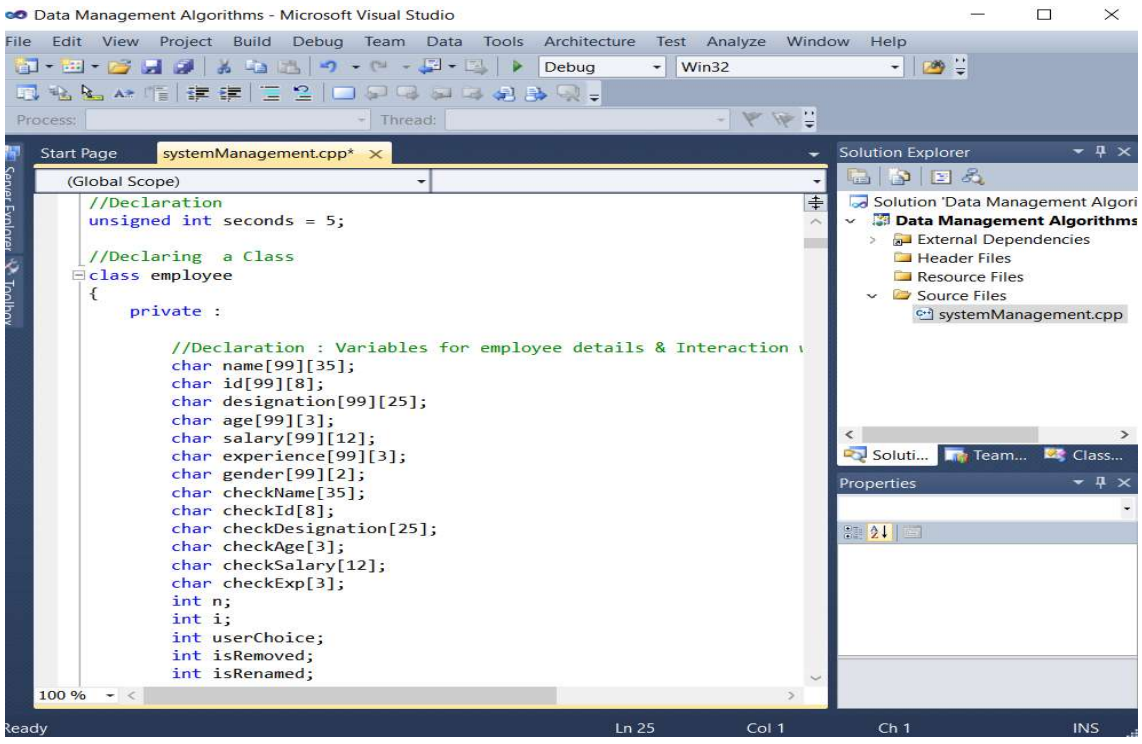
-
-
-

Public:

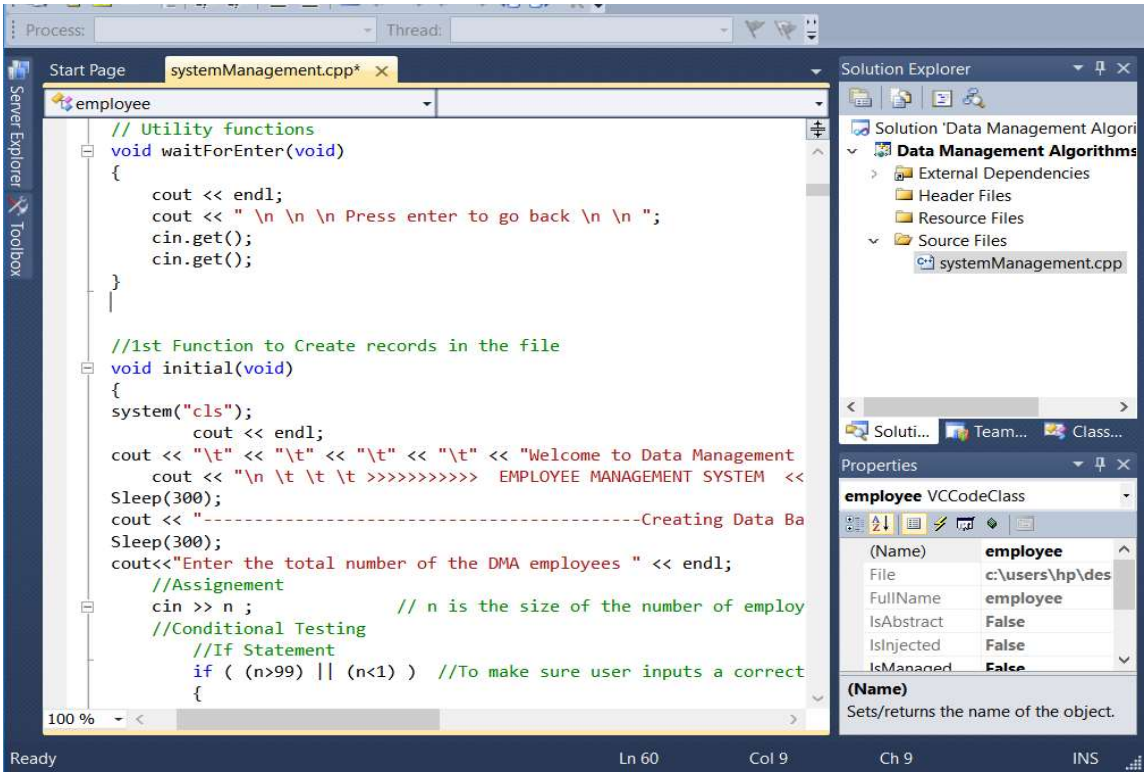
-
-
-



picture a) Header Files

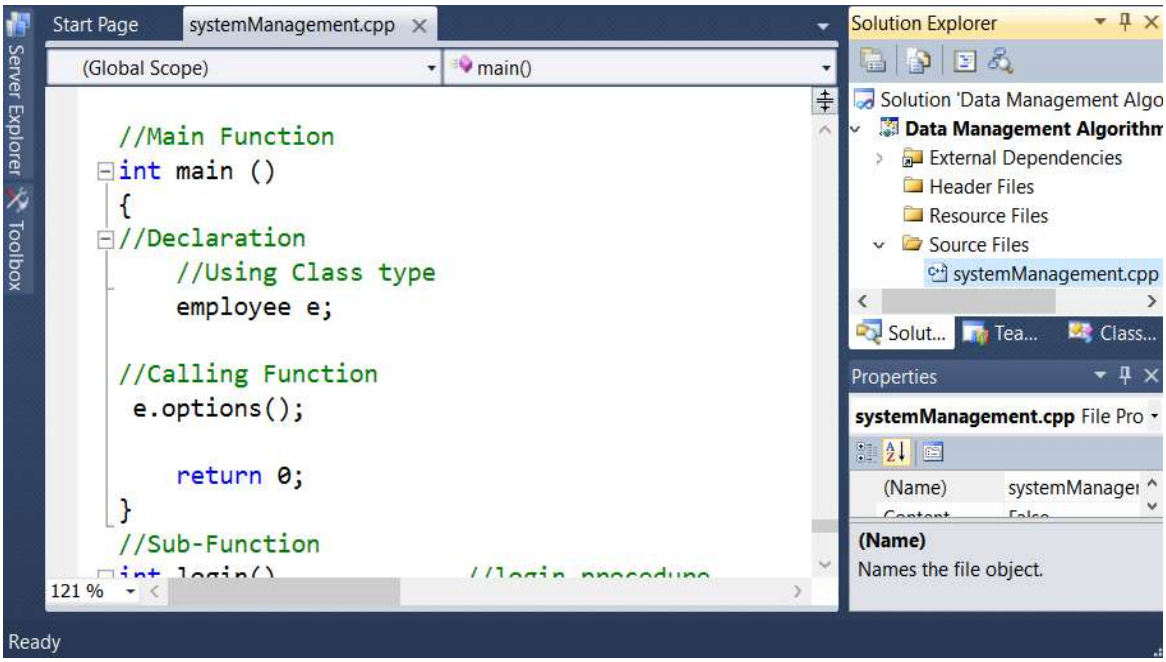


picture b) Variables



picture c) Sub-Functions

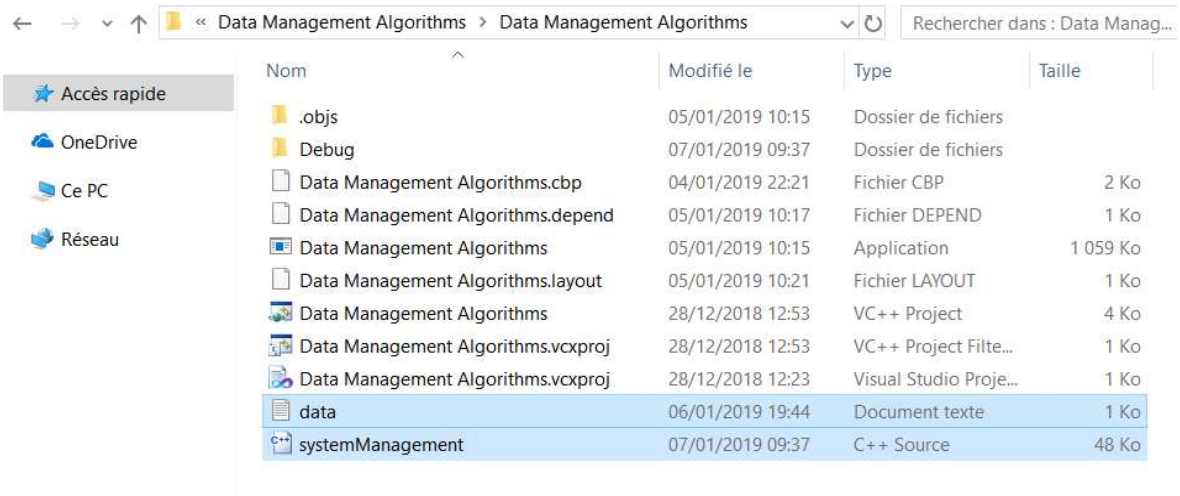
-
-
-



picture d) Main Function

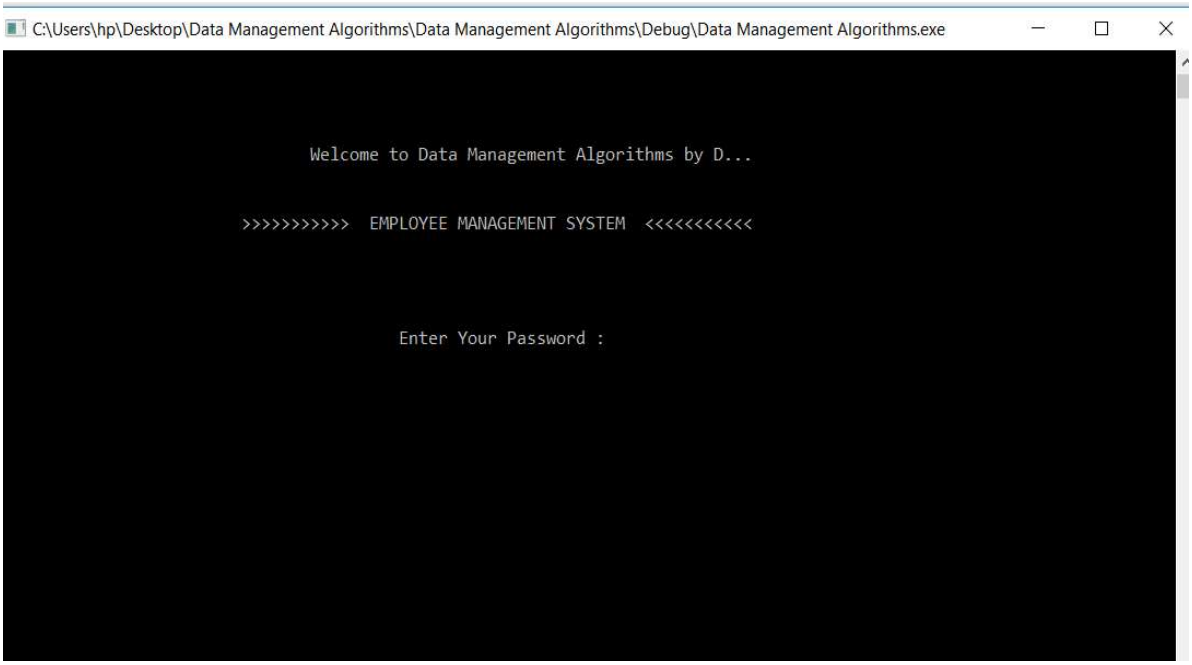
5. RUNNING

(1) Instructions



picture 1: File Instructions

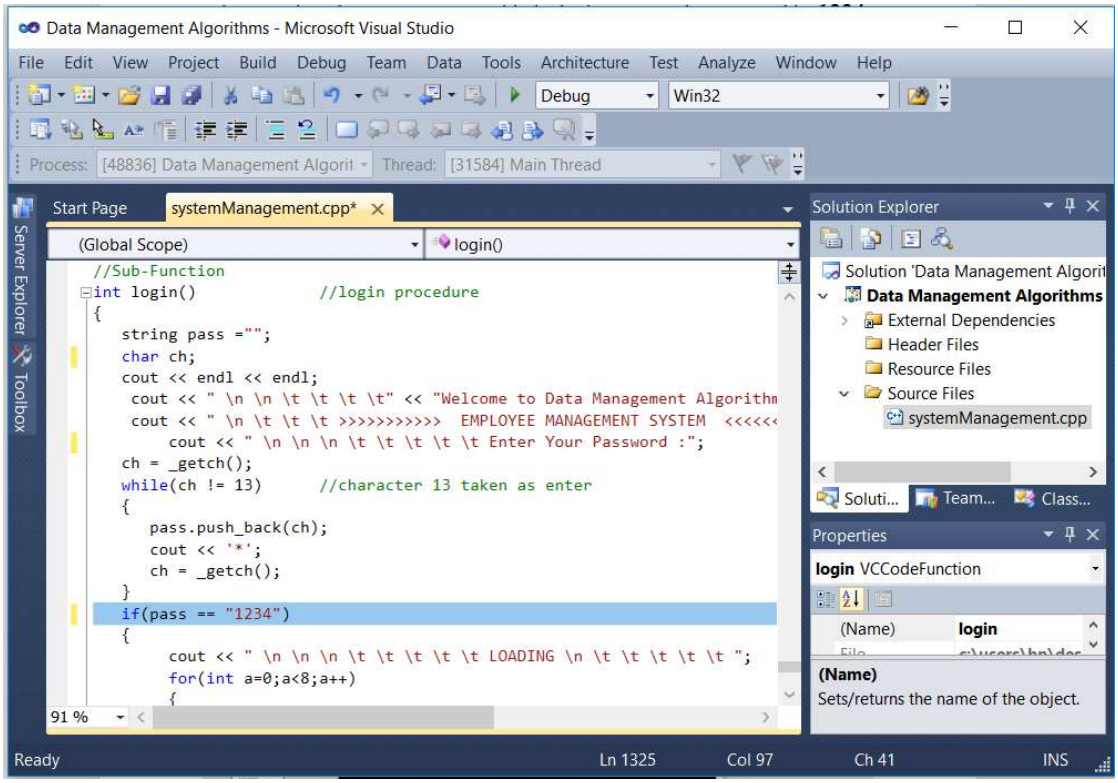
The text file **data (.text)** must be in the project folder with the code source **systemManagement (.cpp)** or else the program will not properly work.



picture 2: Login Process

When running, the program starts with the login process, the password is: **1234.**

And if you wish to change the password: Then, open source file, scroll down until you reach the last sub-function, search for int login()



picture 3: Changing Password

Line 135 you can change the password, replace 1234 by the new password you want (it can be anything) then don't forget **to save and compile**.

* * *

All the records 'datatype is **char** because they are much easier to handle and mostly for the predefined keyword **strcmp** which plays a very big part in the program.

The designation must not exceed 24 characters and must be a no space string

The name must not exceed 34 characters and must be a no space string

The age must not exceed 2 characters (integer only)

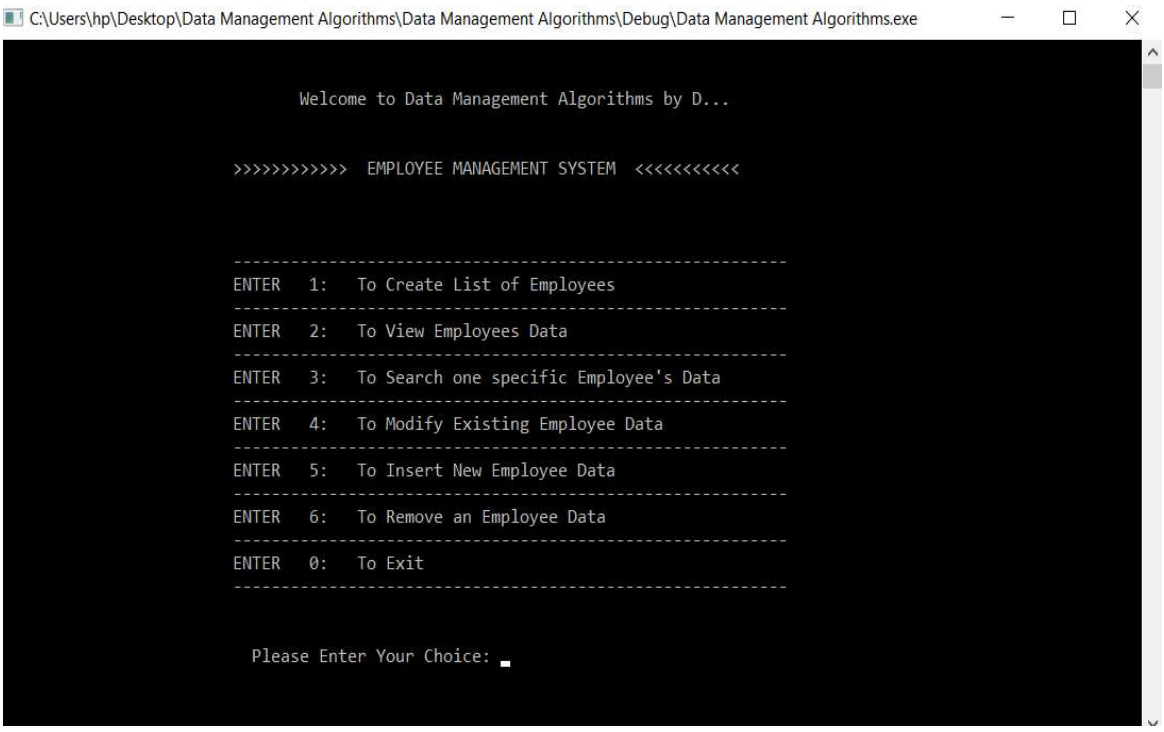
The years of experience must not exceed 2 characters (integer only)

The gender must not exceed 1 character (M or F)

The salary must not exceed 10 characters (integer only)

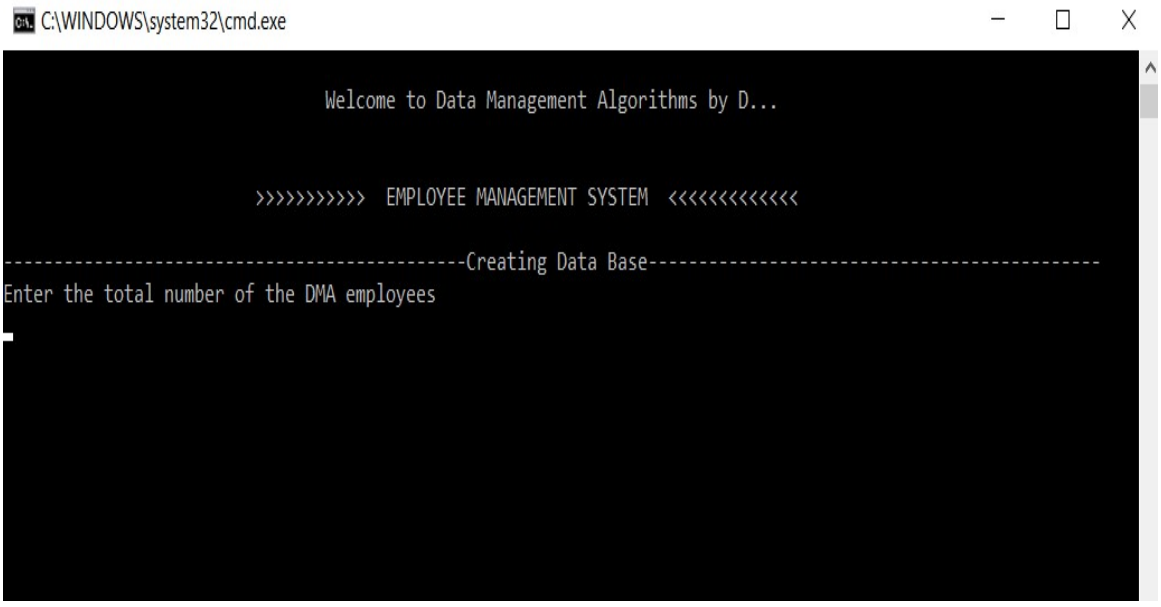
Every employee's records must be different.

(2) Running

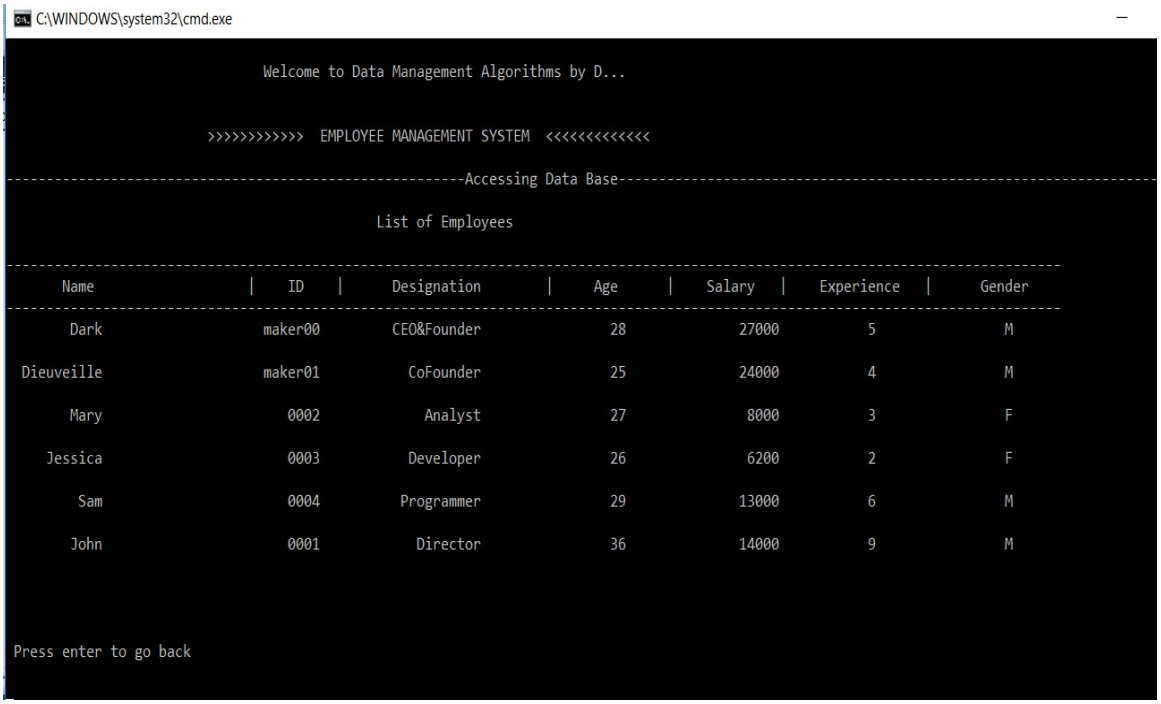


picture 4: Main Menu

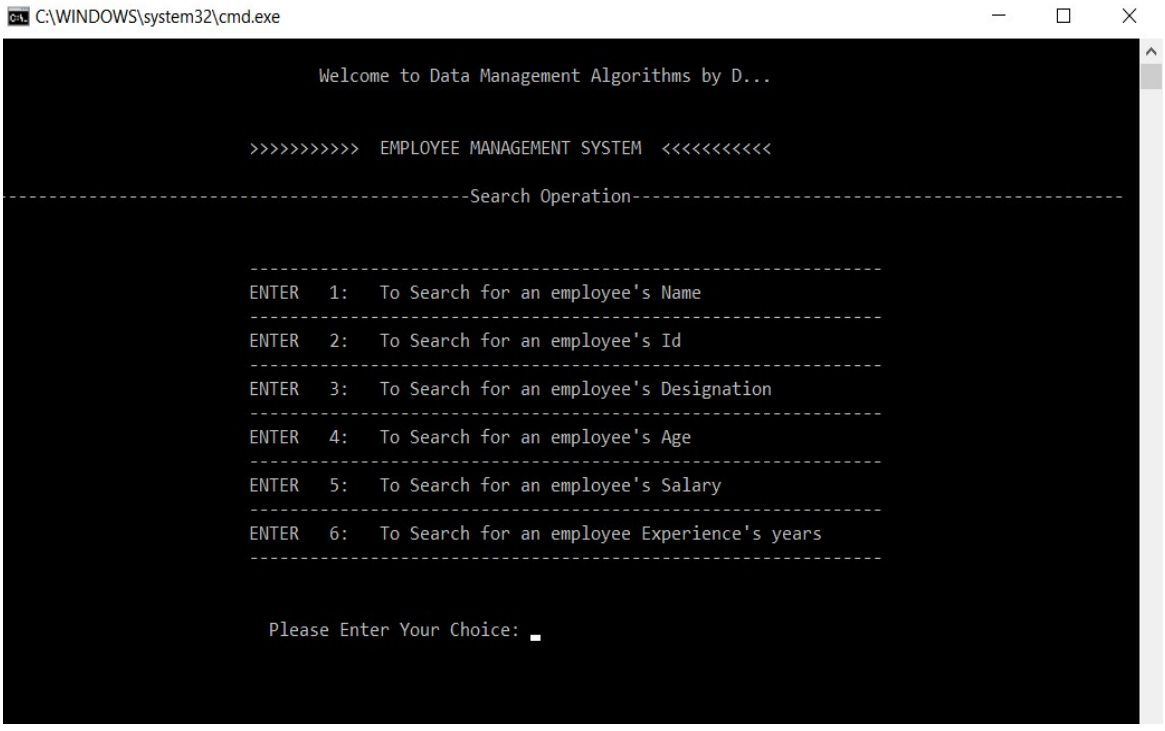
From here, for a better visualization: extend the console application to its full size.



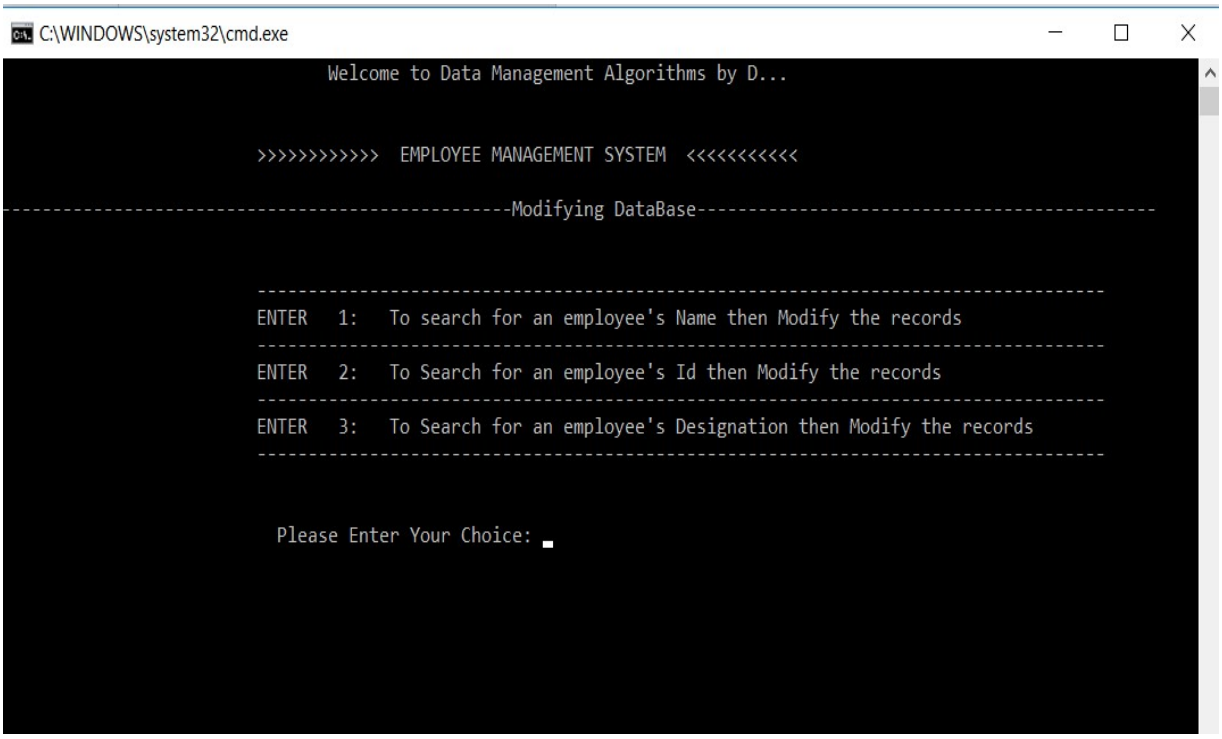
picture 5: Create Operation



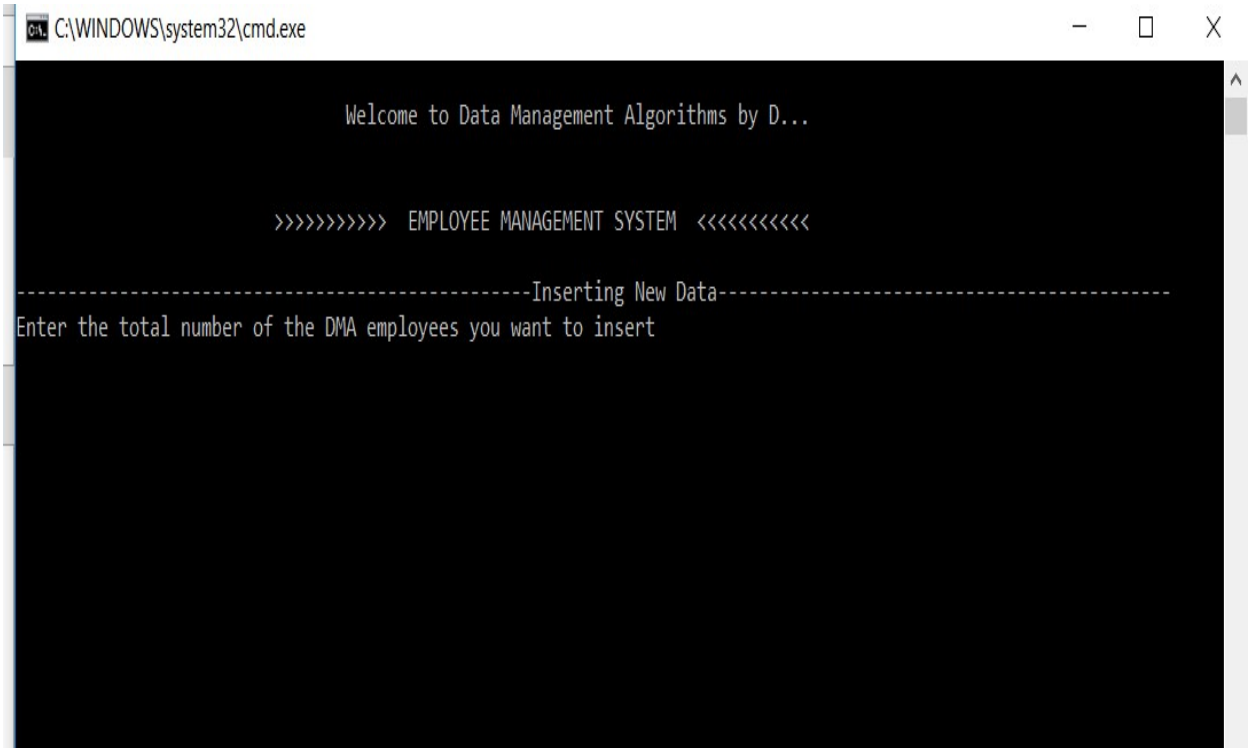
picture 6: Output Operation



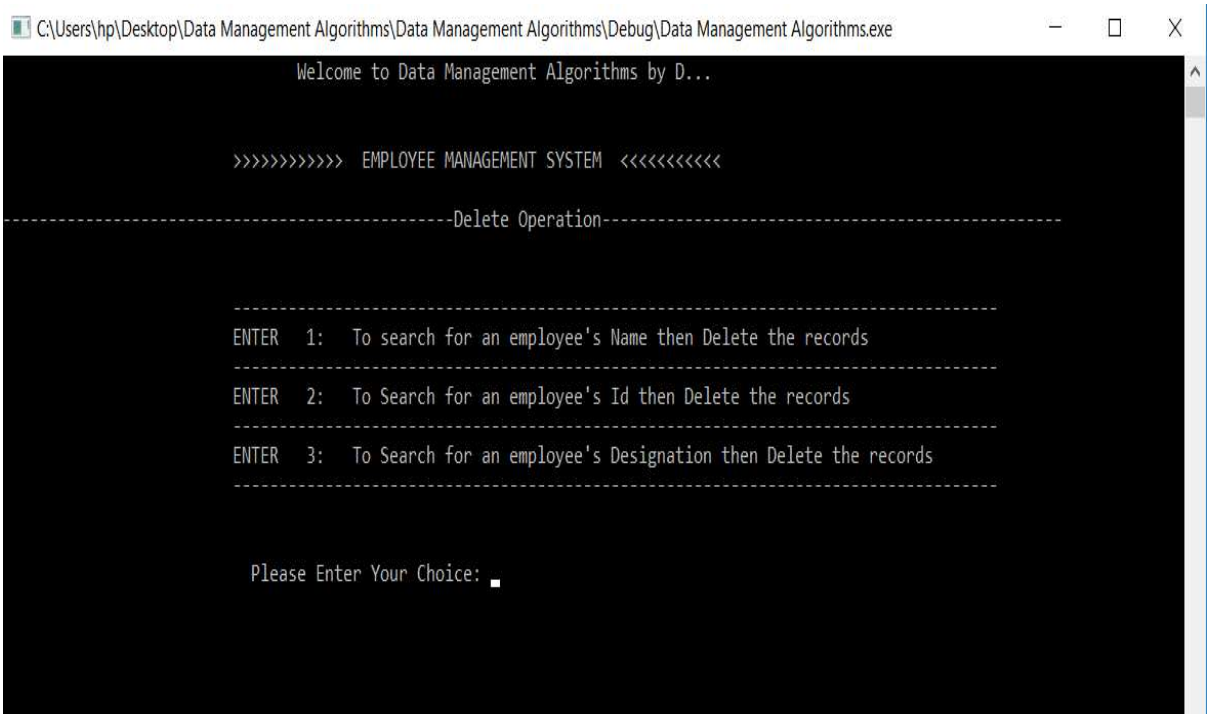
picture 7: Search Operation



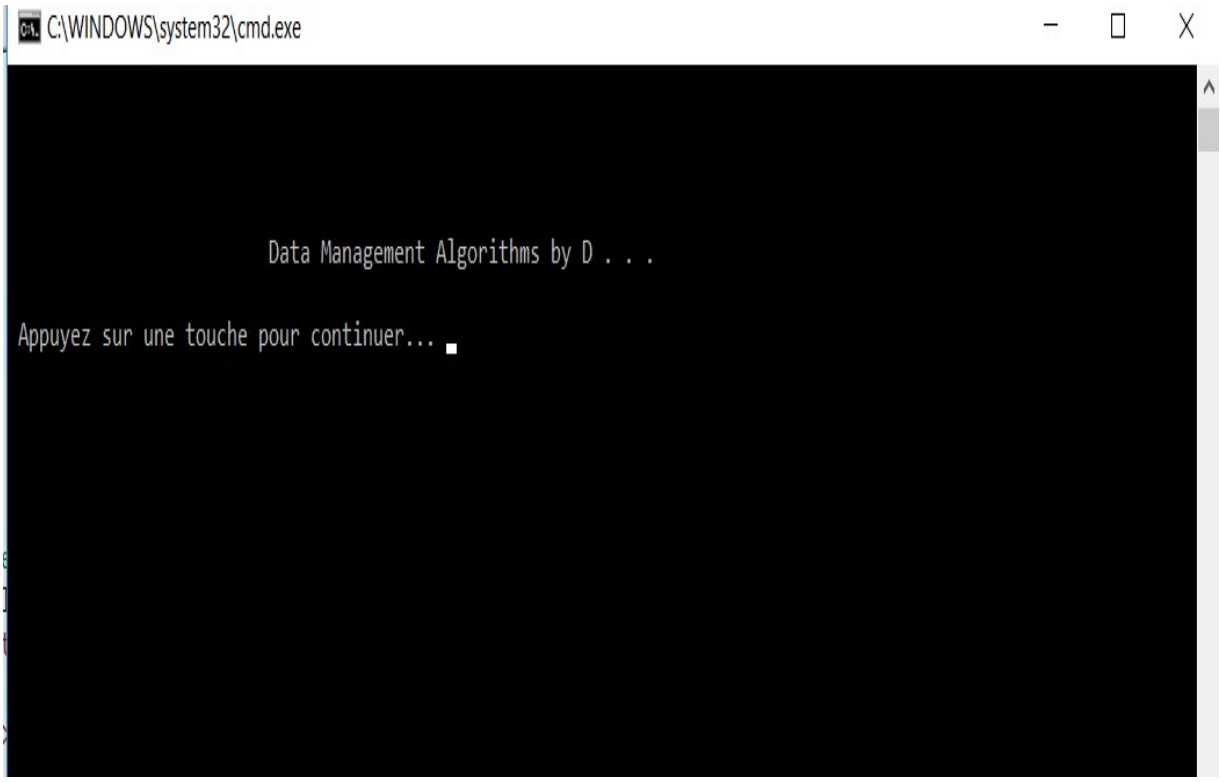
picture 8: Modify Operation



picture 9: Insert Operation



picture 10: Delete Operation



picture 11: Termination

SUMMARY

The DMA (Data Management Algorithms by D …) with its Employee System Management program allows the user to implement 6 operations: Create, Output, Search, Modify, Insert, Delete. It is a very efficient system that store employees 'records: Name, ID, Designation, Salary, Years of Experience and Gender. The user defines the number of employees from 1 up to 99, creates the list by inputting the information, then can view it, also search for a specific employee's information by several methods, then modify or insert the records and even delete them, and all of that with a file saving every changes, the physical structure used is sequential access file, which means the system keeps every record or data even after shutting down the program. Every employee's records must be different.

The system requires from the users to carefully read and follow the instructions.

Also to pay attention to every messages outputted in the screen.

All the records' datatype is char because they are much easier to handle and mostly for the predefined keyword strcmp which plays a very big part in the program. The text file data (.text) must be in the project folder with the code source systemManagement (.cpp) or else the program will not work.