“Employee management system”

A Mini Project

Submitted by

Aayushi Dorle (SC001)

Abhishek Patil (SC002)

Adinath Mahangare (SC003)

SECOND YEAR COMPUTER ENGINEERING

Under the Guidance of

Dr. Ajitkumar Shitole

Department of Computer Engineering

Hope Foundation's
International Institute of Information Technology

Hinjawadi, Pune – 411057

AY 2020-2021
Semester-4

TABLE OF CONTENTS

1. PROBLEM STATEMENT

Write a program in Python/C/C++ to implement Employee Record System which should consist of the following operations:
1. Insert
2. Display
3. Search
4. Modify

2. INTRODUCTION

- ➔ This project is a program to manage employee's records.
- ➔ This program uses the concept of the data storing in the array.
- ➔ The user can add the member's details by entering his/her name, age, employee ID and years of experience.
- ➔ Besides, the user can view a list of all available employees with their details.
- ➔ The user can add, view, search and remove data using file handling.
- ➔ Developed using C language.
- ➔ Easy to operate and understandable.

3. ALGORITHM AND FLOWCHART

- Algorithm:

1. Define struct employee with variables name, code, designation, experience, age.

2. Define int num and function showMenu.

3. Define build function to build a table of employee entries created during the program with specific number of employees.

4. Define display function to display created table of employee entries.

5. Define insert function to insert a new entry at the end of employee database using if and else.

6. Define functions deleteIndex and deleteRecord to delete specific Employee Entry using its Employee ID.

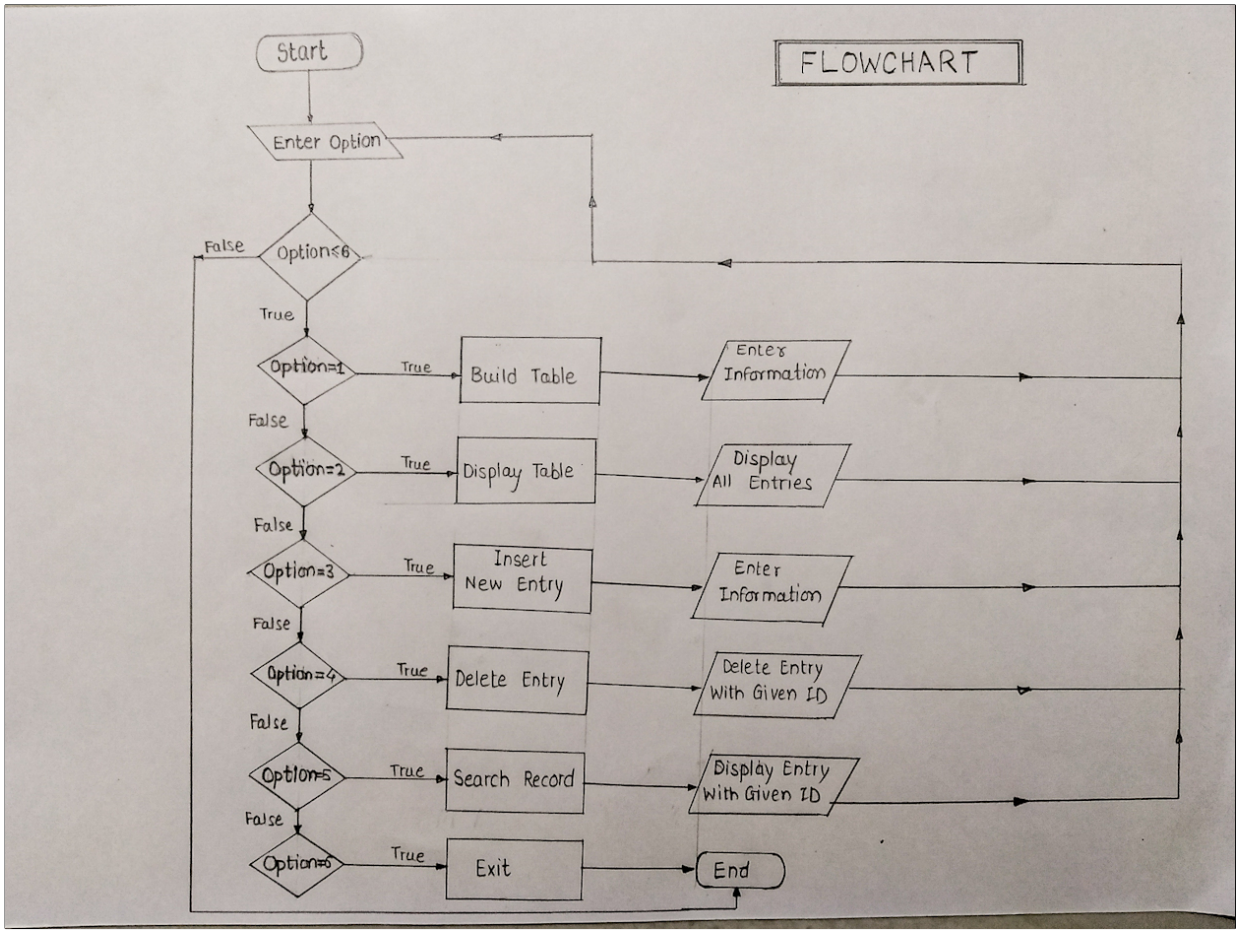
7. Define searchRecord function with linear search method to search information of any Employee using his employee ID.

8. Define showMenu function to display menu with options for which program is compatible.

9. Define int main function and call showMenu function in it.

10. End.

- Flowchart:



4. REQUIREMENTS

4.1. SOFTWARE REQUIREMENTS

Operating System : Microsoft Windows XP or Later or Equivalent
Frontend : Codeblocks
Backend :Turbo C++

4.2. LIBRARIES/PACKAGES USED

- **<stdlib.h>**
  This header defines several general purpose functions, including dynamic memory management, random number generation, communication with the environment, integer arithmetics, searching, sorting and converting.
- **include <stdio.h>**
  Stdio. **h** is known to contain the input and output operations like "printf" or "scanf" etc. "**h**" extension means the header file. If you

request to use a header file in the program by actually **including** it with the preprocessed C directive #**include** like you have seen the inclusion of **stdio**.

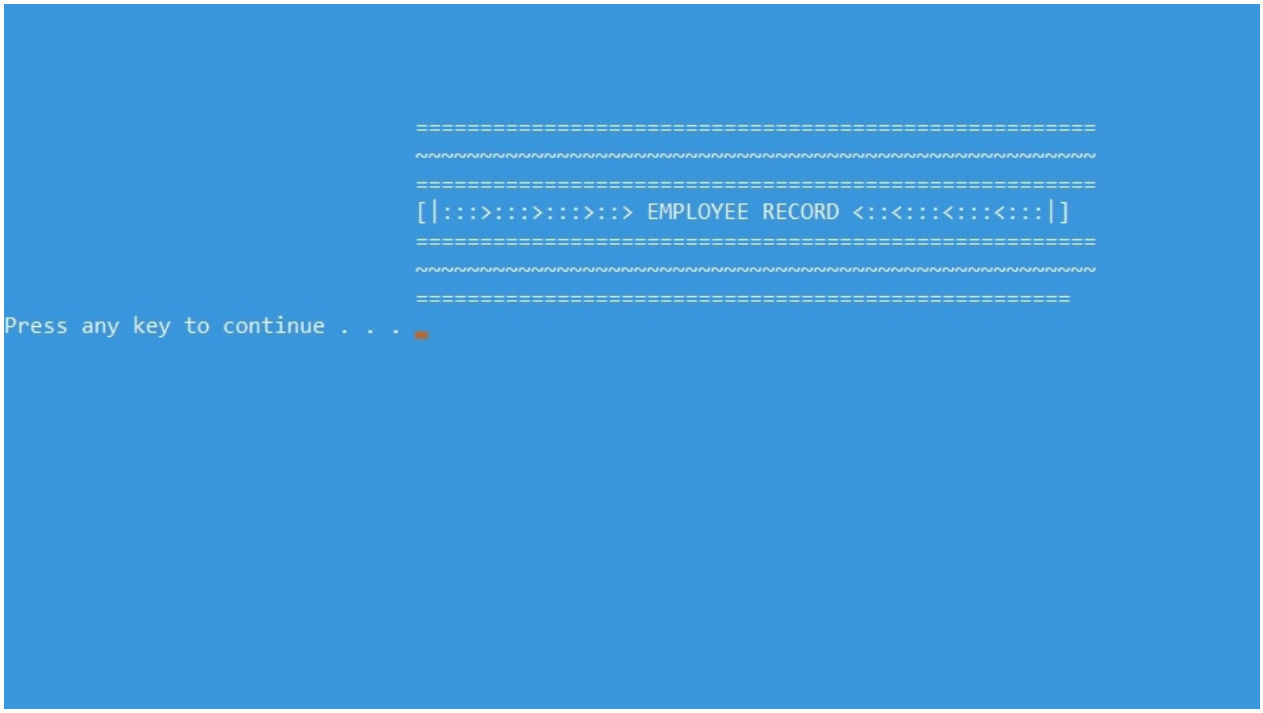
- **<string.h>**
  The string.h header defines one variable type, one macro, and various functions for manipulating arrays of characters.
- **<windows.h>**
  Windows.h is a Windows-specific header file for the C and C++ programming languages which contains declarations for all the functions in the Windows API, all the common macros used by Windows programmers, and all the data types used by the various functions and subsystems. It defines a very large number of Windows specific functions that can be used in C.
- **struct**
  The struct keyword defines a structure type followed by an identifier (name of the structure).
- **arrays**
  they are used for storing multiple values in a single variable, instead of declaring separate variables for each value. To declare an array, define the variable type, specify the name of the array followed by square brackets and specify the number of elements it should store.
- **fseek()**
  The C library function int fseek (FILE *stream, long int offset, int whence) sets the file position of the stream to the given offset.
- **printf()**
  Writes the C string pointed by *format* to the standard output (stdout). If *format* includes *format specifiers* (subsequences beginning with %), the additional arguments following *format* are formatted and inserted in the resulting string replacing their respective specifiers.
- **scanf()**
  Reads data from stdin and stores them according to the parameter *format* into the locations pointed by the additional arguments. The additional arguments should point to already allocated objects of the type specified by their corresponding format specifier within the *format* string.
- **fwrite()**
  Writes an array of *count* elements, each one with a size of *size* bytes, from the block of memory pointed by *ptr* to the current position in the *stream*. The *position indicator* of the stream is advanced by the total number of bytes written. Internally, the function interprets the block pointed by ptr as if it was an array of (size*count) elements of type unsigned char, and writes them sequentially to stream as if fputc was called for each byte.
- **while**
  It is a most basic loop in C programming. While loop has one

control condition, and executes as long the condition is true. The condition of the loop is tested before the body of the loop is executed, hence it is called an entry-controlled loop.

- **rewind()**
  In the C Programming Language, the rewind function sets the file position to the beginning of the file for the stream pointed to by stream. It also clears the error and end-of-file indicators for the stream.

- **fflush()**
  fflush() is typically used for output stream only. Its purpose is to clear (or flush) the output buffer and move the buffered data to console (in case of stdout) or disk (in case of file output stream)
- **pointers**
  A pointer is a variable whose value is the address of another variable, i.e., direct address of the memory location. Like any variable or constant, you must declare a pointer before using it to store any variable address.
- **void**
  When used as a function return type, the void keyword specifies that the function does not return a value. When used for a function's parameter list, void specifies that the function takes no parameters.
- **int**
  short for "integer," is a fundamental variable type built into the compiler and used to define numeric variables holding whole number.
- **cin**
  This statement is the instance of the class istream and is used to read input from the standard input device which is usually a keyboard. The extraction operator(>>) is used along with the object cin for reading inputs.
- **cout**
  This statement is the instance of the ostream class. It is used to produce output on the standard output device which is usually the display screen.
- **int main()**
  it means that our function needs to return some integer at the end of the execution, and we do so by returning 0 at the end of the program. 0 is the standard for the “successful execution of the program”.

5. RESULT


[|:::>:::>:::>::> EMPLOYEE RECORD <::<:::<:::<:::|]
Press any key to continue . . .

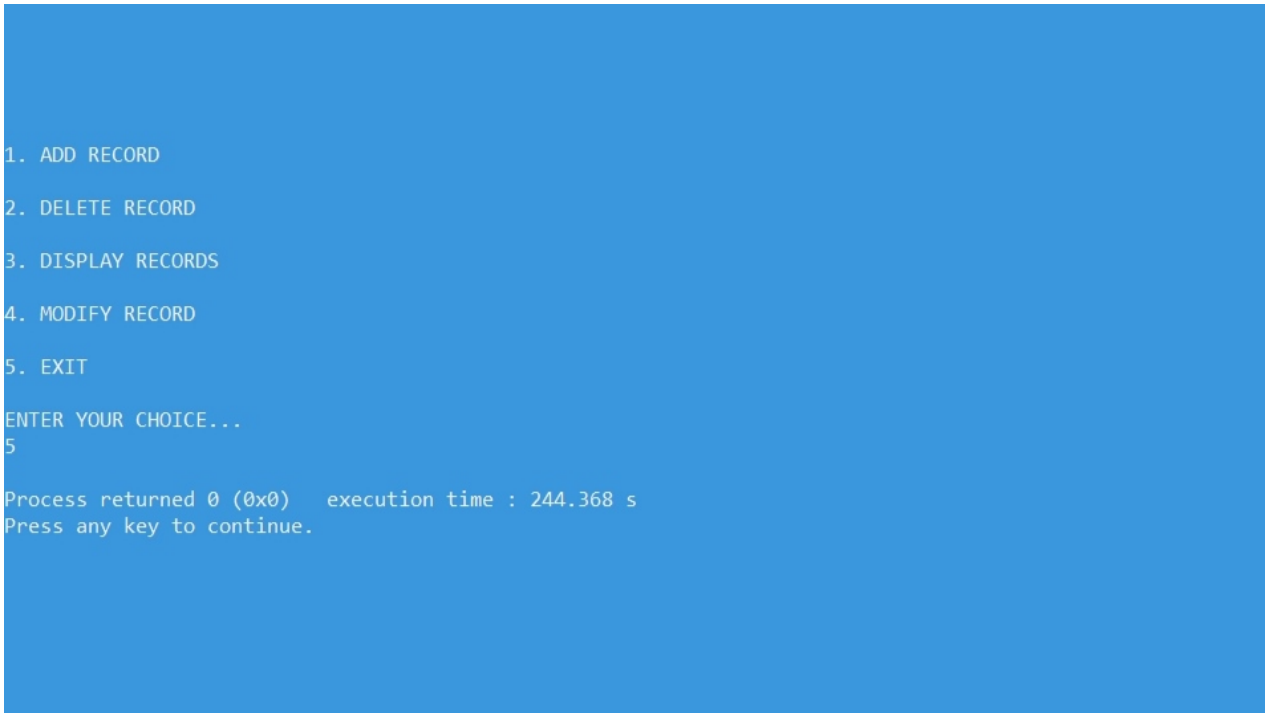

1. ADD RECORD
2. DELETE RECORD
3. DISPLAY RECORDS
4. MODIFY RECORD
5. EXIT
ENTER YOUR CHOICE...

```
Enter Name : Adinath

Enter Age : 20

Enter Salary : 20000

Enter EMP-ID : 1

Want to add another record (Y/N) : y

Enter Name : Ayushi

Enter Age : 20

Enter Salary : 20000

Enter EMP-ID : 2

Want to add another record (Y/N) : y

Enter Name : Abhishek

Enter Age : 20

Enter Salary : 20000

Enter EMP-ID : 3

Want to add another record (Y/N) :
```

```
=============================================================
NAME            AGE             SALARY                  ID
=============================================================

Abhi            21              21000.00                 4
Adinath         20              20000.00                 2
Ayushi          20              25000.00                 1


        Press any key to continue . . .
```

```
Enter employee name to modify : Abhishek

Enter new name:Abhi

Enter new age :21

Enter new salary :21000

Enter new EMP-ID :4

Want to modify another record (Y/N) :
```

```
Enter employee name to delete : Adinath

Want to delete another record (Y/N) :
```



6. CONCLUSION

In this project we have successfully created a program to manage the employees and classify them accordingly, by using the concepts of array and the various kinds of operation performed on it.

7. REFERENCES

- https://www.geeksforgeeks.org/arrays-in-c-cpp/#:~:text=An%20array%20in%20C%20or,using%20indices%20of%20an%20array.&text=They%20can%20be%20used%20to,etc%20of%20any%20particular%20type.
- https://www.tutorialspoint.com/data_structures_algorithms/array_data_structure.htm