



Small Scale Employee Information Management System

In Partial Fulfillment of the requirements in CS106P: Data Structures and Algorithms

Presented by:

RAMON EMMANUEL P. BORNEA, BSIS 2nd

MARC JEREMY U. CEDEÑO, BSIS 2nd

JHON LOUISE S. TAN, BSIS 3rd

Presented to:

PROF. MARTZEL P. BASTE

October 10, 2022

I. Introduction

A. Problem Scenario

A small-scale organization, in a rural area, with a staff size of 20. They use manual and paper records to keep track of their data and require a new system to manage their employees. The problem is that other organizations have already adopted a digital system for managing the information of their employees, and the organization is left out, adding more hassle work for its managers.

Paper records can be tiered, or the documents are flammable, and accessing data is more time-consuming because managers need to browse the record, which takes up more time.

The organization also considered the budget of the basic employee management system and decided it was time to implement an employee management system that would make their workflow more fluid and have a more efficient procedure for keeping their employees information.

B. Objectives

For a small-scale organization to transfer from manual tracking of personal information of their employee to a more advanced method using technology, the program will have the following objectives:

1. Keep track of personal details of employees in the organization.
2. Change from manual employment management system to a digital management system

C. Project Definition

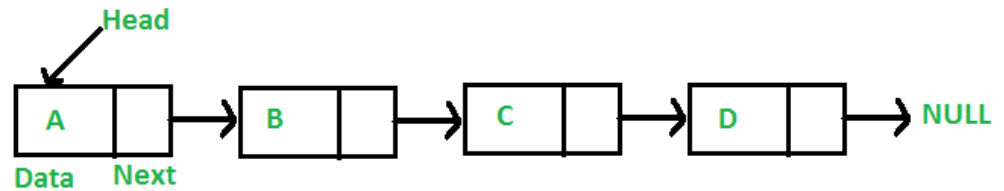


Figure 1. Linked list

This employee management system uses linked list linear data structure that adds, removes, searches, and edits a specific node. As seen in figure 1, the head is considered the first node of the list. In a linked list, data are not stored in neighboring memory locations. Each node has data and a pointer pointing to the next data of the next node (*Linked List Data Structure*, 2022). However, for our system, each node consists of the employee's name, age, phone number, and assigned ID number.

D. Time and Space Complexity

In computer science, the idea of time complexity refers to the quantification of how long a set of instructions or an algorithm will take to process or execute in relation to the volume of input. In other terms, the length of time it takes for a program to process an input is its temporal complexity. An algorithm's effectiveness is determined by two factors: time complexity and space complexity (*Time Complexities of Different Data Structures*, 2022).

Time complexity is defined as how many times a specific instruction set is executed as opposed to how long it takes in total. The reason for this is that, while space complexity is the entire amount of memory

the program needs to run, time complexity also depends on external factors like the compiler used, processor speed, etc. (*Time Complexities of Different Data Structures*, 2022).

Below is the time complexity of the linked list application.

Algorithm	Average Case	Worst Case
Add	$O(1)$	$O(1)$
Delete	$O(1)$	$O(1)$
Search	$O(N)$	$O(N)$
Edit	$O(N)$	$O(N)$



II. Project/System Prototyping

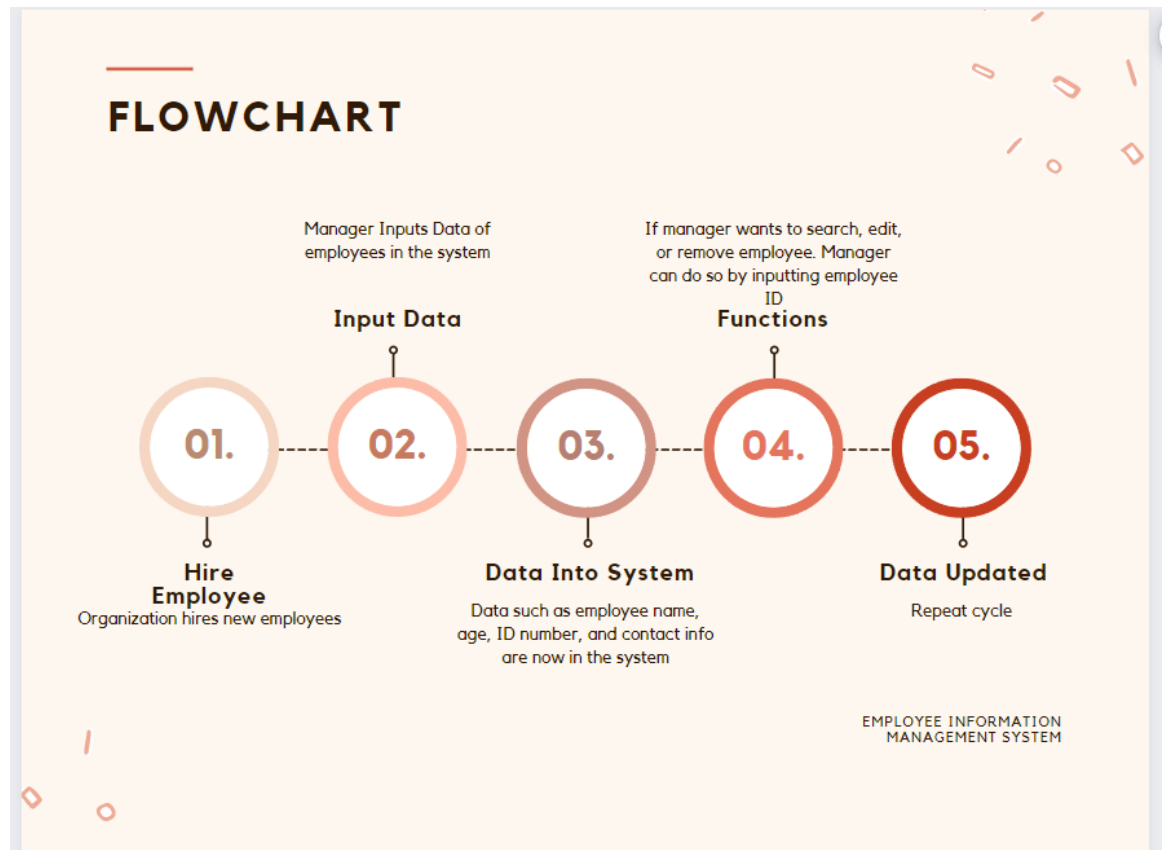


Figure 2. Small Scale Employee Information Management System Flowchart

Figure 2 shows the flowchart for the small scale employee management system. When new employees are hired into the organization, the manager will input their data into the system and fill out the information needed such as employee name, employee ID number, age and contact info.

After the data is filled out, the data is now stored in the program system. When the manager or any higher positions wants to access the information about a specific employee, they can access it by inputting the employee ID number in the search feature. The information then can be edited and

updated. If an employee leaves the organization, the manager can then remove the employee data by first searching for the employee ID number and then removing the data from the system.

III. Sample Input/Output

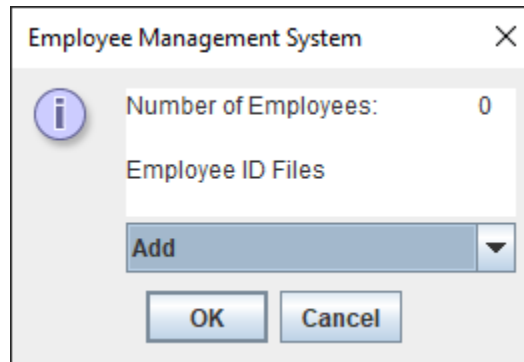


Figure 3. Employee Information Management System Main Interface

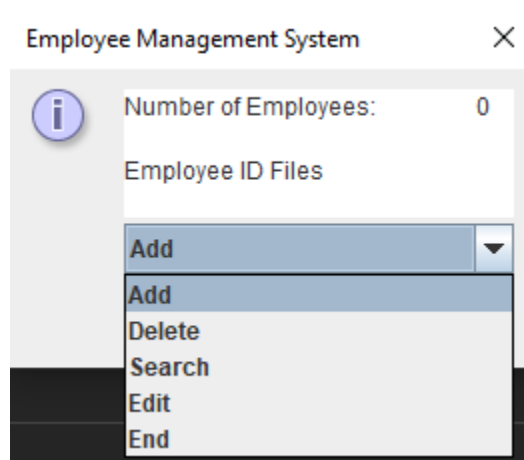


Figure 4. Employee Information Management System Main Interface (with options)

After running the program, the user will be greeted with a small window application. It displays the number of employees that has a file and the text files of the employees that have been entered. The user will have five options: add, delete, search, edit, and end.

Input ✕


 Enter name

Figure 5. Adding Information (input name)

Input ✕


 Enter age

Figure 6. Adding Information (input age)

Input ✕


 Enter contact number

Figure 7. Adding Information (input contact number)

Input ✕


 Enter ID number

Figure 8. Adding Information (input ID number)

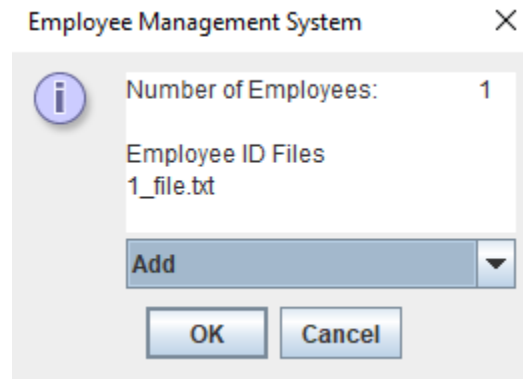


Figure 9. Main interface after adding

When using the add function, the user must input the employee's name, age, contact number, and assigned ID number for them. After inputting each parameter, the program will create a text file and save it to a directory the user chooses. The main interface will show an updated number of employees counter and display the text files that has been created. It is important to change the directory path in the code for this interface to work.

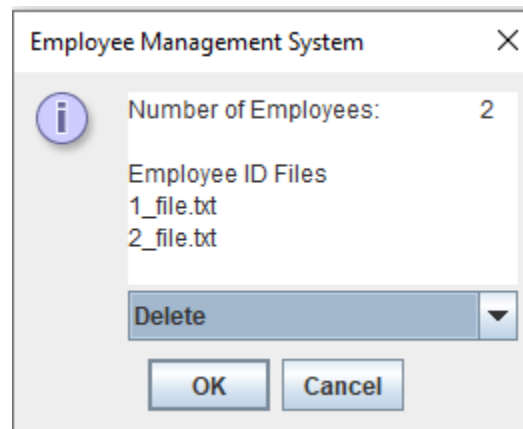


Figure 10. Delete option

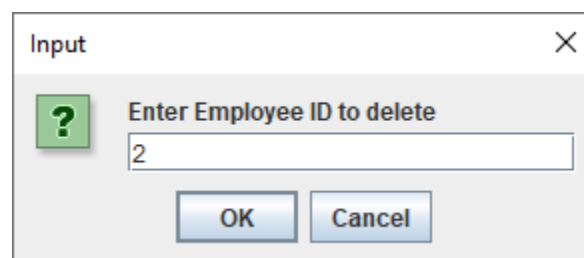


Figure 11. Delete text file (input ID number)

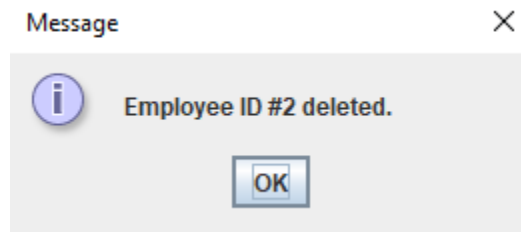


Figure 12. A prompt to show the file has been successfully deleted

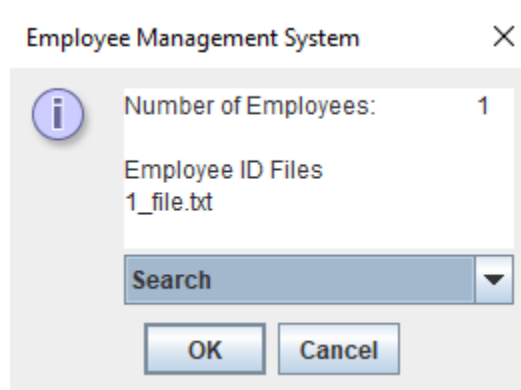


Figure 13. Main interface after deleting

When the user picks the delete option, the program will need the ID number of that employee. It will search for the ID number entered by the user and search for it in the database. When it matches, it will delete the file and will display a prompt indicating that the file has been deleted successfully.

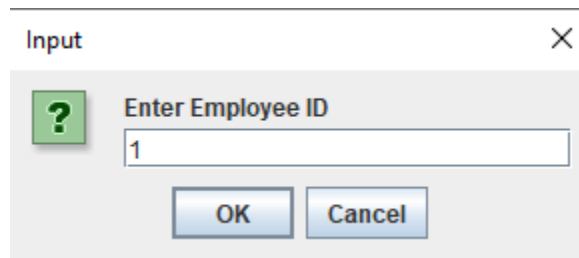


Figure 14. Searching (input ID number)

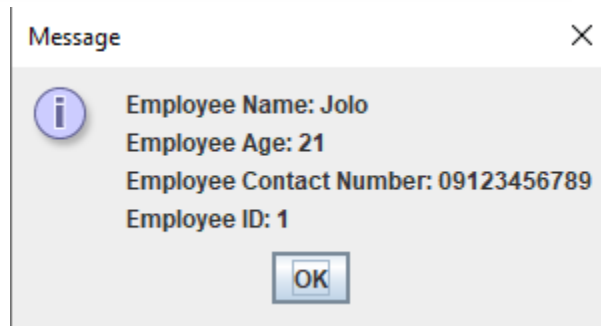


Figure 15. Searching (displays all information entered)

When the user wants to look up all of an employee's information, they can simply use the search function. They will only need to input the ID number of that employee and once the program matches the input with the database, it will display all of the necessary information.

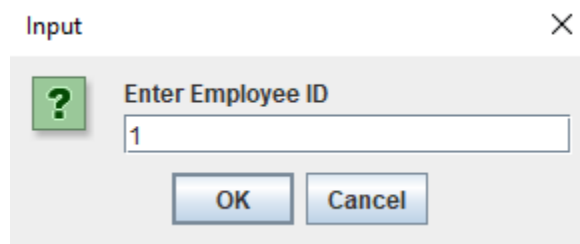


Figure 16. Editing (input ID number)

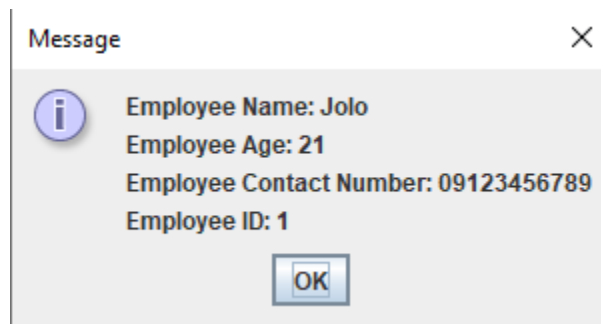


Figure 17. Editing (show all information before editing)

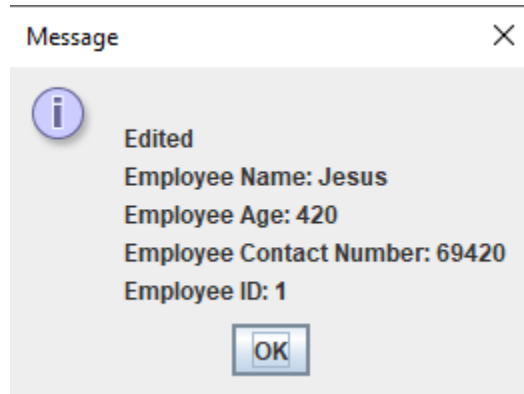


Figure 18. Employee ID #1's information after editing

Finally, when the user wants to edit a specific employee's information, they can use the edit function. Like the other functions, it asks for the ID number of that employee. Once it matches with the database, it will display all of the information to double check if they got the right employee or what information they want to change. After editing, the user can simply search for that employee ID and it will display the newly edited information.

IV. Source Code

A. Employee Class

```
import javax.swing.JOptionPane;
import java.util.*;
import java.io.*;

public class Employee{
    private String name,number;
    private int age,id;
```



```
public Employee head,next;

public Employee(){
    name="";
    age=0;
    number="";
    id=0;
    next=head=null;
}

public Employee(String name, int age, String
number, int id){
    this.name=name;
    this.age=age;
    this.number=number;
    this.id=id;
    next=head=null;
}

public void error_message(String msg){
JOptionPane.showMessageDialog(null,msg,"Error",JOptionP
ane.ERROR_MESSAGE);
}

    /**Returns <b> true <b> if the list is empty (head
is equal to null)
    * @return true or false
    */
public boolean isEmpty(){
    return head==null;
}

    /**
    * Returns the size of the list
    * @return int
    */
```



```
public int currentSize(){
    int ctr=0;
    File path=new File("D:/Jolo/Programming
Shit/Data Structures and Algorithms"); // change the
path to match your system's
    File[] files=path.listFiles(new
FilenameFilter(){
        @Override
        public boolean accept(File dir, String
name){
            return name.endsWith(".txt");
        }
    });
    for (File file:files){
        ctr++;
    }
    return ctr;

    // int counter=1;
    // if(isEmpty()){
    //     return 0;
    // }else{
    //     Employee link=head;
    //     while(link.next!=null){
    //         link=link.next;
    //         counter++;
    //     }
    //     return counter;
    // }

}

/**
 * Checks if the Employee ID is unique
 * @param id - int
 * @return boolean
 */
public boolean uniqueID(int id){
```



```
Employee link;
link=head;
while(link.id!=id){
    link=link.next;
    return true;
}
return false;
}

/**
 * Displays the Employee ID and name
 * @return String
 */
public String display(){
    String hold="";
    File path=new File("D:/Jolo/Programming
Shit/Data Structures and Algorithms"); // change the
path to match your system's
    File[] files=path.listFiles(new
FilenameFilter(){
        @Override
        public boolean accept(File dir, String
name){
            return name.endsWith(".txt");
        }
    });
    for (File file:files){
        hold+=file.getName()+"\n";
    }
    return hold;
    // if(isEmpty()){
    //     return "";
    // }else{
    //     String hold="";
    //     Employee link=head;
    //     while(link!=null){
    //         hold+=link.id+"\t"+link.name+"\n";
    //     }
    // }
```



```
//          link=link.next;
//      }
//      return hold;
//  }
}

// public String display(Employee n){
//     String hold="";
//     while(n!=null){
//         hold+=n.id+"\t"+n.name+"\n";
//         n=n.next;
//     }
//     return hold;
// }

/**
 * Adds data
 * @param name - String
 * @param age - int
 * @param number - String
 * @param id - int
 * @return void
 */
public void add(String name, int age, String
number, int id){
    Employee newNode=new Employee();
    try{
        if(isEmpty()){
            File f1=new File(id+"_"+file.txt");
            if(f1.createNewFile()){
                newNode.name=name;
                newNode.age=age;
                newNode.number=number;
                newNode.id=id;
                newNode.next=null;
                head=newNode;
                FileWriter writer=new
FileWriter(id+"_"+file.txt");
```



```
        writer.write("Employee Name:
"+newNode.name+
                                "\nEmployee Age:
"+newNode.age+
                                "\nEmployee Contact
Number: "+newNode.number+
                                "\nEmployee ID:
"+newNode.id);
        writer.close();
    }
    }else{
        if(uniqueID(id)==true){
            File f1=new
File(id+"_ "+"file.txt");
            if(f1.createNewFile()){
                newNode.name=name;
                newNode.age=age;
                newNode.number=number;
                newNode.id=id;
                newNode.next=head;
                head=newNode;
                FileWriter writer=new
FileWriter(id+"_ "+"file.txt");
                writer.write("Employee Name:
"+newNode.name+
                                "\nEmployee Age:
"+newNode.age+
                                "\nEmployee Contact
Number: "+newNode.number+
                                "\nEmployee ID:
"+newNode.id);
                writer.close();
            }
        }else{
            error_message("Please enter a
unique ID number");
        }
    }
}
```




```
    }  
    } catch (Exception e) {  
  
    }  
}  
  
/**  
 * Deletes the data for the selected employee ID  
 * @param id - int  
 * @return void  
 */  
// public void deleteAtID(int id) {  
//     if (isEmpty()) {  
//         error_message("List is empty.");  
//     } else if (id < 0) {  
//         error_message("Employee not found.");  
//     } else {  
//         Employee link, visit;  
//         visit = link = head;  
//         if (link == null) {  
//             error_message("Employee not found");  
//         }  
//         if (link != null && link.id == id) {  
//             head = link.next;  
//         }  
//         while (link != null && link.id != id) {  
//             visit = link;  
//             link = link.next;  
//         }  
//         visit.next = link.next;  
//         JOptionPane.showMessageDialog(null,  
"Employee ID #" + id + " deleted.");  
//     }  
// }  
  
public void deleteAtID(int id) {  
    File file = new File(id + "_" + "file.txt");
```



```
// if(isEmpty() && !file.exists()) {  
//     error_message("List is empty");  
// }  
  
if(file.exists()) {  
    if(file.delete()) {  
        // Employee link, visit;  
        // visit = link = head;  
        // if(link != null && link.id == id) {  
        //     head = link.next;  
        // }  
        // while(link != null && link.id != id) {  
        //     visit = link;  
        //     link = link.next;  
        // }  
        // visit.next = link.next;  
        JOptionPane.showMessageDialog(null,  
"Employee ID #" + id + " deleted.");  
    }  
} else {  
    error_message("Employee not found");  
}  
}  
  
/**  
 * Gets data of selected employee ID  
 * @param id - int  
 * @return String  
 */  
public String search(int id) throws Exception {  
    // String hold = "";  
    // Employee link;  
    // link = head;  
    // if(link == null) {  
    //     return "List is empty";  
    // } else if(id < 0) {  
    //     return "Employee not found.";  
    // }
```



```
// while(link!=null&&link.id!=id) {  
//     link=link.next;  
// }  
// hold="Employee Name: "+link.name+  
//     "\nEmployee Age: "+ link.age+  
//     "\nEmployee Contact Number:  
"+link.number+  
//     "\nEmployee ID: "+link.id;  
// return hold;  
File file = new File(id+"_"+"file.txt");  
Scanner sc = new Scanner(file);  
String hold="";  
while (sc.hasNextLine())  
{  
    hold+=sc.nextLine()+"\n";  
}  
return hold;  
}  
  
/**  
 * This method edits the already entered data  
 * @param code - int  
 * @param name - String  
 * @param age - int  
 * @param number - String  
 * @param id - int  
 * @return void  
 */  
public void edit(int code,String name, int age,  
String number, int id) throws IOException{  
    Employee link;  
    link=head;  
    if(link==null){  
        error_message("Employee not found");  
    }  
  
    while(link!=null&&link.id!=code){
```



```
        link=link.next;
    }
    link.name=name;
    link.age=age;
    link.number=number;
    link.id=id;
    FileWriter writer=new
FileWriter(id+"_"+"file.txt");
    writer.write("\nEdited\nEmployee Name:
"+link.name+
                "\nEmployee Age: "+link.age+
                "\nEmployee Contact Number:
"+link.number+
                "\nEmployee ID: "+link.id);
    File file=new File(id+"_"+"file.txt");
    File rename=new File(link.id+"_"+"file.txt");
    file.renameTo(rename);
    writer.close();
}
}
```

B. EmployeeMenu Class

```
import javax.swing.JOptionPane;
import javax.swing.JTextArea;

public class EmployeeMenu{
    public static void main(String[] args){
        String
menu[]={ "Add", "Delete", "Search", "Edit", "End" };
        String hold="", choice="";
        String name="", number="";
        int age=0, id=0;
        Employee emp=new Employee();
    }
}
```



```
do{
    hold="Number of
Employees:\t"+emp.currentSize()+
        "\n\nEmployee ID Files"+
        "\n"+emp.display();
    choice=JOptionPane.showInputDialog(null,
new JTextArea(hold),"Employee Management
System",1,null,menu,menu[0]).toString();
    switch(choice){
        case "Add":
            try{
name=JOptionPane.showInputDialog(null, "Enter name");
age=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter age"));
number=JOptionPane.showInputDialog(null, "Enter contact
number");
id=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter ID number"));
                emp.add(name, age, number, id);
            }catch(Exception e){
JOptionPane.showMessageDialog(null, "Please enter a
valid input","Error",JOptionPane.ERROR_MESSAGE);
            }//end of try catch
            break;
        case "Delete":
id=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter Employee ID to delete"));
                emp.deleteAtID(id);
            break;
        case "Search":
            try{
```



```
id=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter Employee ID"));

JOptionPane.showMessageDialog(null,emp.search(id));
    }catch(Exception e){

JOptionPane.showMessageDialog(null, "Please enter a
valid input","Error",JOptionPane.ERROR_MESSAGE);
    }//end of try catch
    break;
    case "Edit":
        try{
            int
code=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter Employee ID"));

JOptionPane.showMessageDialog(null,emp.search(code));

name=JOptionPane.showInputDialog(null, "Enter name");

age=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter age"));

number=JOptionPane.showInputDialog(null, "Enter contact
number");

id=Integer.parseInt(JOptionPane.showInputDialog(null,
"Enter ID number"));

            if(emp.uniqueID(id)==true){
                emp.edit(code, name, age,
number, id);
            }else{
                emp.error_message("Please
enter a unique ID number");
            }
        }catch(Exception e){
```



```
JOptionPane.showMessageDialog(null, "Please enter a  
valid input","Error",JOptionPane.ERROR_MESSAGE);  
        }//end of try catch  
        break;  
        case "End":  
        }//end of switch  
        }while(!choice.equals("End"));//end of do  
    }//end of main  
}//end of class
```

V. Curriculum Vitae



COLLEGE OF COMPUTER
AND INFORMATION SCIENCE
MAPÚA MALAYAN COLLEGES MINDANAO



Ramon Emmanuel P. Bornea

CONTACT



09165376948



rebornea@mcm.edu.ph



Phase 1 block 8 lot 32,
Elenita heights, Catalunan
Grande, Davao City
Philippines

SKILLS

Python programming

Java programming

Communication skills

LANGUAGES

ENGLISH

TAGALOG

HOBBIES

Video Games

Hiking

Cycling

[Hi! I'm Ramon. I like doing activities that induces adrenaline. May it be cycling, hiking, or from playing chess. I like to do thing that stimulates my brain. I'm a person who is eager to explore more of what the world has to offer.

Before enrolling to this course, I had no experience in programming, but I was willing to learn more about it since I also love computers. This curiosity has led to my exponential growth and passion for programming in java and python.

EDUCATION

B.S Information Systems 2nd year-present

Malayan Colleges Mindanao

Senior High School 2019

Malayan Colleges Mindanao

EXPERIENCE

Student

Malayan Colleges Mindanao

No experience

present



Marc Jeremy U. Cedeño

CONTACT



09770595880



mjCedeno@mcm.edu.ph



Block 10 Lot 30, Cedar
Street, Prescilla Estate 2,
Cabantain, Davao City,
Philippines

SKILLS

Beginner Learning Java

Beginner at Programming

LANGUAGES

ENGLISH

TAGALOG

HOBBIES

Reading
Dancing
Martial Arts
Gym
Gaming

[Hi! I'm Marc. I love reading books about my interest such as mangas and books, I am an active person based on my hobbies. I also like playing video games occasionally. I have an Introverted personality and I like trying new things that Interest me.

As a beginner, I am currently trying to catch up in learning Java as I have no background or experience in programming before enrolling in Information Systems.

EDUCATION

B.S Information Systems 2nd year present

Malayan Colleges Mindanao

Clubs & Societies: Malayan Dance Company

Senior High School 2020

Malayan Colleges Mindanao

Clubs & Societies: Malayan Dance Company

EXPERIENCE

Student

Malayan Colleges Mindanao

No experience yet

present



Jhon Louise S. Tan

CONTACT



09164756972



jltan@mcm.edu.ph



Valma Subdivision, Purok
Malakas, Barangay San
Isidro, Gerenal Santos
City, Philippines

SKILLS

Java Programming

Python Programming

LANGUAGES

ENGLISH

Tagalog

HOBBIES

Video Games
Reading Manga

[Hi! I'm Jolo. I love playing competitive video games, GeoGuessr, and online chess. I also love consuming Japanese media like music, anime, and manga. I am a curious person and I learning more about what piques my curiosity and interest.

Before enrolling in IS, I had no experience or background in programming. Then as time passed, I learned to love programming. Currently, I only know how to write programs using Java and Python.

EDUCATION

B.S Information Systems 3rd year present

Malayan Colleges Mindanao

Clubs & Societies: CCIS Department

Senior High School 2019

The Quantum Academy inc.

EXPERIENCE

Student

Malayan Colleges Mindanao

No experience yet

present

References

- ;. (2020, November 26). ; - Wiktionary. Retrieved October 10, 2022, from <https://codezilla.com/download/employee-management-system-in-java-with-source-code/>
- Admin. (2020, April 4). *Importance, Benefits, & Types Of Employee Management System*. EmpMonitor. Retrieved October 10, 2022, from <https://empmonitor.com/blog/employee-management-system>
- Employee Information Management System, Employee Information Management Software Services, Employee Information Management Software Development and App Development – Snovasy.com*. (2020, November 26). Snovasy.com. Retrieved October 10, 2022, from <https://www.snovasy.com/products/employee-information-management-system/>
- Kanchev, K. D. (2006, December). Employee Management System. <https://lnu.diva-portal.org/smash/get/diva2:204828/FULLTEXT01.pdf>
- Lascano, C. (2021, August 29). *Do small companies need an Employee Management System?* Manila Standard. Retrieved October 10, 2022, from <https://manilastandard.net/business/business-columns/green-light/363574/do-small-companies-need-an-employee-management-system.html>
- Linked List Data Structure*. (2022, June 18). GeeksforGeeks. Retrieved October 10, 2022, from <https://www.geeksforgeeks.org/data-structures/linked-list/>
- Time complexities of different data structures*. (2022, June 13). GeeksforGeeks. Retrieved October 10, 2022, from <https://www.geeksforgeeks.org/time-complexities-of-different-data-structures/>



COLLEGE OF COMPUTER
AND INFORMATION SCIENCE

MAPÚA MALAYAN COLLEGES MINDANAO