EMPLOYEE PAYROLL SYSTEM

PROJECT BASED LEARNING

This code is the implementation of a Payroll Management System in C++. It includes classes and functions for managing employee records, such as adding new employees, displaying employee details, modifying records, and listing all employees. Below is a brief overview of the code:

 # CLASSES DETAILED EXPLANATION

1. `LINES` Class:

- This class is responsible for handling various display-related functions and drawing lines and boxes on the screen.

Member Functions:

- `LINE_HOR(int column1, int column2, int row, char c)`: Draws a horizontal line with the specified character 'c' from 'column1' to 'column2' at the given row.

- `LINE_VER(int row1, int row2, int column, char c)`: Draws a vertical line with the specified character 'c' from 'row1' to 'row2' at the given column.

- `BOX(int column1, int row1, int column2, int row2, char c)`: Draws a rectangular box with the specified character 'c' enclosing the area defined by 'column1', 'row1', 'column2', and 'row2'.

- `CLEARUP()`: Clears the screen from the current cursor position to the top of the screen.

- `CLEARDOWN()`: Clears the screen from the current cursor position to the bottom of the screen.

2. `MENU` Class:

- This class is responsible for displaying menus and handling menu navigation in the Payroll Management System.

Member Functions:

- `MAIN_MENU()`: Displays the main menu options and handles user input to navigate between different functionalities of the system.

- `EDIT_MENU()`: Displays the edit menu options (used for modifying or deleting records) and handles user input within the edit menu.

- `INTRODUCTION()`: Displays an introductory message or information about the Payroll Management System.

3. `EMPLOYEE` Class:

- This class represents an employee in the system and manages employee-related operations, such as adding, modifying, deleting, displaying, listing, and generating salary slips.

Member Functions:

- `ADD_RECORD(...)`: Adds a new employee record to a data file with the provided information.

- `MODIFY_RECORD(...)`: Modifies an existing employee record in the data file with the provided information.

- `DELETE_RECORD(...)`: Deletes an employee record from the data file based on the employee code.

- `LASTCODE()`: Retrieves the code of the last employee record in the data file.

- `FOUND_CODE(...)`: Checks if an employee with the given code exists in the data file.

- `RECORDNO(...)`: Retrieves the record number of an employee based on their code.

- `DISPLAY_RECORD(...)`: Displays the details of an employee with the specified code.

- `LIST()`: Lists all employee records with basic information (code, name, phone, date of joining, designation, grade, and salary).

- `NEW_EMPLOYEE()`: Handles the process of adding a new employee by collecting and validating their information.

- `DISPLAY()`: Displays detailed information about a specific employee.

- `MODIFICATION()`: Handles the process of modifying an employee's information.

- `DELETION()`: Handles the process of deleting an employee's record.

- `SALARY_SLIP()`: Generates a salary slip for an employee, if applicable.

#DATA MEMBERS

This C++ code is the program for a Payroll Management System. It allows users to perform various tasks related to employee management and payroll processing. Here's an explanation of the key data members and functions in the code:

1. **Class Definitions:**

- `LINES`: This class appears to be responsible for drawing lines, boxes, and clearing the screen.

- `MENU`: The `MAIN_MENU` and `EDIT_MENU` functions are part of this class and are used to display the main menu and edit menu options to the user.

- `EMPLOYEE`: This class represents an employee and contains various member functions to perform operations on employee records.

2. **Data Members in EMPLOYEE Class:**

- `code`: Employee code or ID.

- `dd`, `mm`, `yy`: Employee's date of joining (day, month, year).

- `name`, `address`, `phone`: Employee's name, address, and phone number.

- `desig`: Employee's designation.

- `grade`: Employee's grade (A, B, C, D, E).

- `house`, `convense`: Flags indicating if the employee receives house allowance and conveyance allowance.

- `loan`: Loan amount for the employee.

- `basic`: Basic salary of the employee.

3. **Member Functions in EMPLOYEE Class:**

- `ADD_RECORD`: Adds a new employee record to the file.

- `MODIFY_RECORD`: Modifies an existing employee record.

- `DELETE_RECORD`: Deletes an employee record.

- `LASTCODE`: Retrieves the last employee code used.

- `FOUND_CODE`: Checks if a given employee code exists.

- `RECORDNO`: Retrieves the record number for a given employee code.

- `LIST`: Displays a list of all employees.

- `DISPLAY_RECORD`: Displays the details of a specific employee.

- `NEW_EMPLOYEE`: Collects information to create a new employee record.

- `DISPLAY`: Displays the details of a specific employee.

- `MODIFICATION`: Allows the user to modify an existing employee record.

4. **Menu Functions in MENU Class:**

- `MAIN_MENU`: Displays the main menu with options to manage employees, display records, and more.

- `EDIT_MENU`: Displays an edit menu for options like deleting or modifying employee records.

5. **Drawing Functions in LINES Class:**

- `LINE_HOR`: Draws a horizontal line.

- `LINE_VER`: Draws a vertical line.

- `BOX`: Draws a box with specified characters.

- `CLEARUP` and `CLEARDOWN`: Clear the screen above and below a certain point.

Overall, this program offers functionalities for managing employee records, including adding, modifying, deleting, and displaying employee information. It also provides options for generating salary slips and displaying lists of employees.

 # MEMBER FUNCTIONS

The `EMPLOYEE` class represents an employee in the Payroll Management System and manages various operations related to employees. Below are the member functions of the `EMPLOYEE` class:

1. `ADD_RECORD(...)`: Adds a new employee record to a data file with the provided information.

- This function takes various parameters such as the employee's code, name, address, phone number, date of joining, designation, grade, and basic salary.

- It opens the data file for writing, appends the new employee's details to the end of the file, and then closes the file.

2. `MODIFY_RECORD(...)`: Modifies an existing employee record in the data file with the provided information.

- This function takes the employee's code and the new information for the employee (e.g., name, address, phone number, etc.) as parameters.

- It first checks if an employee with the given code exists in the data file using the `FOUND_CODE(...)` function.

- If the employee is found, it opens the data file for reading and writing, locates the employee's record, updates it with the new information, and then closes the file.

3. `DELETE_RECORD(...)`: Deletes an employee record from the data file based on the employee code.

- This function takes the employee's code as a parameter.

- It checks if an employee with the given code exists in the data file using the `FOUND_CODE(...)` function.

- If the employee is found, it opens the data file for reading and writing, locates the employee's record, deletes it, and then closes the file.

4. `LASTCODE()`: Retrieves the code of the last employee record in the data file.

- This function opens the data file, seeks to the end of the file, reads the code of the last employee, and then closes the file.

- It is often used to generate unique employee codes when adding a new employee.

5. `FOUND_CODE(...)`: Checks if an employee with the given code exists in the data file.

- This function takes the employee's code as a parameter.

- It opens the data file, searches for the code among existing employee records, and returns a boolean value indicating whether the code was found.

6. `RECORDNO(...)`: Retrieves the record number of an employee based on their code.

- This function takes the employee's code as a parameter.

- It opens the data file and finds the record number (position) of the employee with the given code.

7. `DISPLAY_RECORD(...)`: Displays the details of an employee with the specified code.

- This function takes the employee's code as a parameter.

- It opens the data file, locates the employee's record using the code, and then displays all the details of the employee.

8. `LIST()`: Lists all employee records with basic information (code, name, phone, date of joining, designation, grade, and salary).

- This function opens the data file, reads all employee records, and displays basic information about each employee.

9. `NEW_EMPLOYEE()`: Handles the process of adding a new employee by collecting and validating their information.

- This function interacts with the user to input the details of the new employee, validates the input, and then calls `ADD_RECORD(...)` to add the employee to the data file.

10. `DISPLAY()`: Displays detailed information about a specific employee.

- This function interacts with the user to input an employee code and then calls `DISPLAY_RECORD(...)` to display the employee's details.

11. `MODIFICATION()`: Handles the process of modifying an employee's information.

- This function interacts with the user to input an employee code and the new information, validates the input, and then calls `MODIFY_RECORD(...)` to update the employee's record.

12. `DELETION()`: Handles the process of deleting an employee's record.

- This function interacts with the user to input an employee code, validates the input, and then calls `DELETE_RECORD(...)` to delete the employee's record.

13. `SALARY_SLIP()`: Generates a salary slip for an employee, if applicable.

- This function interacts with the user to input an employee code, retrieves the employee's record, calculates the salary based on the grade and basic salary, and then displays the salary slip.

#OBJECT ORIENTED APPROACH

1. **Class Definition (`EMPLOYEE` Class)**:

- The code defines a class named `EMPLOYEE`. In OOP, a class is a blueprint for creating objects. In this case, the `EMPLOYEE` class defines the structure and behavior of an employee, including their attributes (member variables) and operations (member functions).

2. **Encapsulation**:

- The `EMPLOYEE` class encapsulates data by declaring private member variables (e.g., `empCode`, `empName`, etc.). This encapsulation hides the internal details of an employee's data from the outside world, ensuring that these variables can only be accessed and modified through the defined member functions.

3. **Abstraction**:

- The member functions of the `EMPLOYEE` class provide an abstract interface for interacting with employee records. These functions abstract away the underlying implementation details and provide a clear way to perform operations on employee data (e.g., adding, modifying, deleting, listing records).

4. **Data Hiding**:

- Private member variables in the `EMPLOYEE` class are not directly accessible from outside the class. This data hiding ensures that the internal state of an employee object can only be modified through controlled methods (member functions).

5. **Member Functions as Behavior**:

- Member functions within the `EMPLOYEE` class represent the behavior associated with employee records. For example, the `ADD_RECORD()` function adds a new employee record, and the `DISPLAY_RECORD(int)` function displays the details of a specific employee. This aligns with the OOP concept of methods or functions as the behavior of objects.

6. **Object Creation and Usage**:

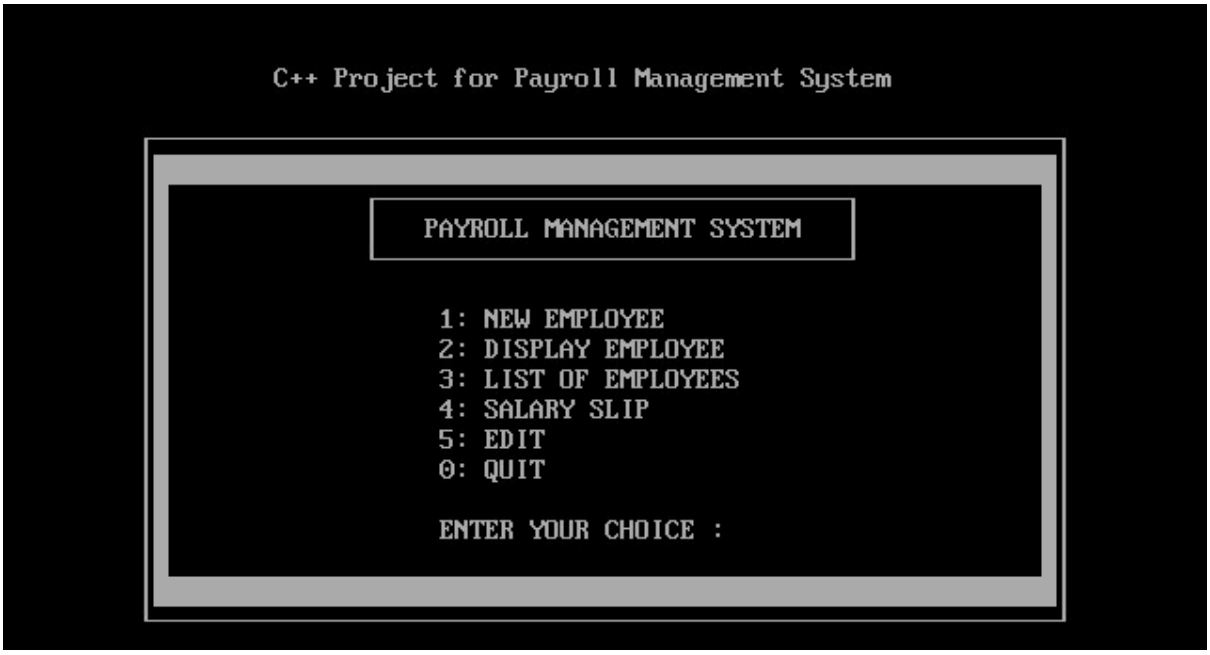
- In a complete program using this code, you would create objects of the `EMPLOYEE` class to represent individual employees. For example:

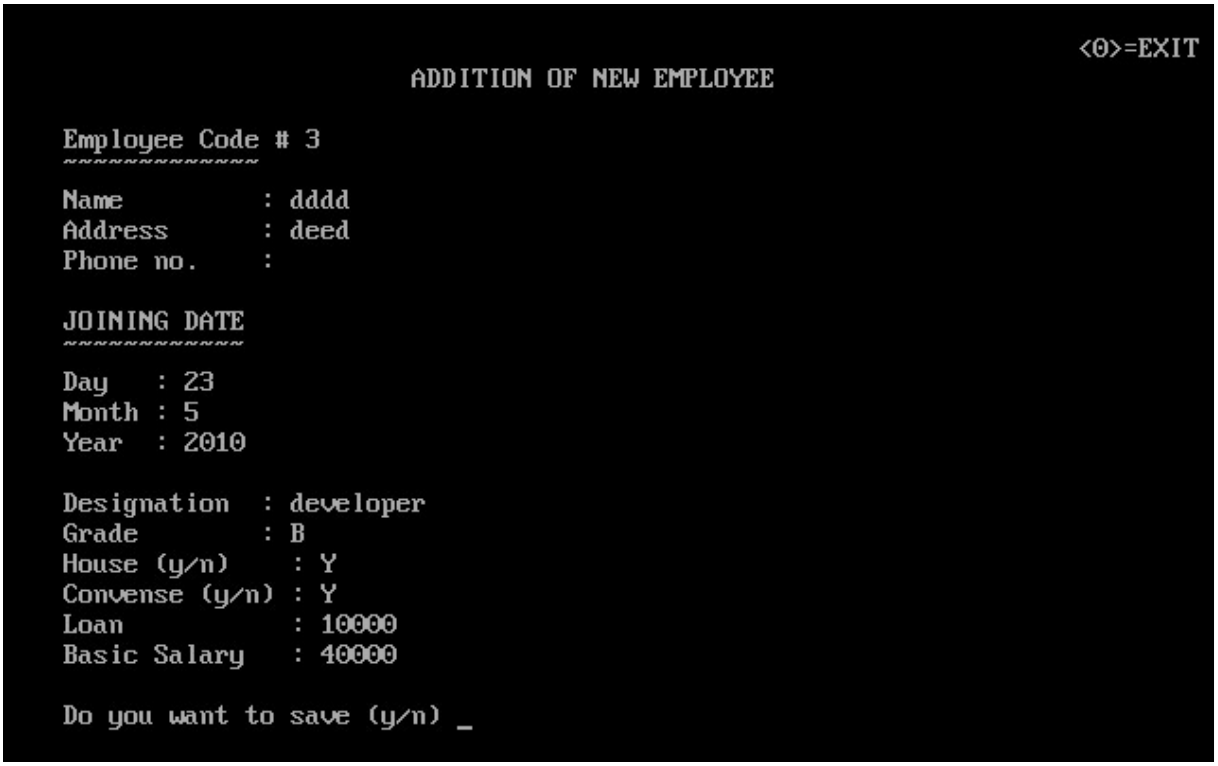
```
EMPLOYEE employee1;

employee1.ADD_RECORD(); // Adding a new employee record using the object
```

7. **Modularity and Reusability**:

- The code defines a reusable `EMPLOYEE` class, allowing you to create multiple instances of this class to manage different employee records. This modularity simplifies code maintenance and promotes reusability.

PROGRAM GUI & OUTPUT





```
Employee Code # 2
~~~~~~~~~~~~~~~

Name          : NNNN
Address       : NMNM
Phone no.     : -

JOINING DATE
~~~~~~~~~~~~

Day   : 9
Month : 9
Year  : 1999

Designation  : MMM
Grade        : D
House (y/n)    : Y
Convense (y/n) : Y
Loan           : 2300
Basic Salary   : 50000


Press any key to continue...
```

```
                          LIST OF EMPLOYEES
                          ~~~~~~~~~~~~~~~~~

CODE NAME                      PHONE    DOJ         DESIGNATION    GRADE  SALARY
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 1   null                      null     0/0/0       null             n    0
 1   AAAA                      -        23/8/2015   BB               C    40000
 2   NNNN                      -        9/9/1999    MMM              D    50000
 3   DDDD                      -        23/5/2010   DEVELOPER        B    40000

Press any key to continue...
```