🧩 Algorithm 1: add_employee()

1. **Start**
2. Declare a variable emp of type struct employee.
3. Open a file named "employee_database.txt" in **append mode** ("a").
4. If the file fails to open:
    - Display "Error opening file!"
    - Stop and return to the main menu.
5. Prompt the user to enter:
    - Employee ID
    - Employee Name
    - Employee Salary
6. Read the inputs and store them in emp.id, emp.name, and emp.salary.
7. Write the employee data to the file using:
   fprintf(file, "%d %.2f %s\n", emp.id, emp.salary, emp.name);
8. Close the file.
9. Display "Employee record added successfully!".
10. **Stop**

🧩 Algorithm 2: display_employees()

1. **Start**
2. Declare a variable emp of type struct employee.
3. Open "employee_database.txt" in **read mode** ("r").
4. If the file cannot be opened:
    - Display "No employee records found!"
    - Stop and return to main menu.
5. Display a table header:
   "ID\tName\t\tSalary"
6. Use a loop to read employee data from the file using:
   fscanf(file, "%d %f %[^\n]", &emp.id, &emp.salary, emp.name)
    - Continue reading until **end of file (EOF)** is reached.
7. For each record read, display:
    - Employee ID

- Employee Name
- Employee Salary

8. Close the file.
9. **Stop**

🧩 **Algorithm 3: main()**

1. **Start**
2. Display program title: "*** Employee Database ***".
3. Create an infinite loop (while(1)) to display the menu repeatedly.
4. Display menu options:
5. 1. Add Employee Record
6. 2. Display Employee Records
7. 3. Exit
8. Ask the user to **enter their choice** and read it in choice.
9. Use a switch statement to handle the menu:
    - **Case 1:** Call add_employee()
    - **Case 2:** Call display_employees()
    - **Case 3:** Display "Exiting the program." and **terminate**.
    - **Default:** Display "Invalid choice! Please try again."
10. Repeat the loop until the user selects Exit.
11. **Stop**