# Experiment 1.1

**Student Name: ROSH UID: 22BAI..**

**Branch: CSE Section:**

**Semester: 6th DOP:**

**Subject: Java Subject Code:22CSH-359**

**Aim:** Create an application to save employee information using arrays.

**Objective:** To develop a functional application that effectively utilizes arrays to store, manage, and retrieve employee information, enabling efficient data organization and manipulation within the application.

# Algorithm:

## Step 1: Initialize the Program

* Start the program.
* Define an array of structures to store employee information.
* Each structure will include fields such as Employee ID, Name, Age, and Department.

## Step 2: Define Functions

1. **Add Employee Information**:
   * Prompt the user to enter details for an employee (ID, Name, Age, Department).
   * Store the entered details in the next available position in the array.
   * Check for array overflow (i.e., maximum number of employees).

## Display All Employee Information:

* + Iterate through the array and print all stored employee details.
  + Handle cases where no employees are stored.

## Search for an Employee:

* + Prompt the user to enter the Employee ID.
  + Search the array for a matching ID.
  + Display the employee's details if found, otherwise print a message indicating the ID is not found.

## Exit Application:

* + Provide an option to exit the program.

## Step 3: Display Menu

* Display a menu with options to:
  1. Add Employee
  2. View All Employees
  3. Search for an Employee
  4. Exit

## Step 4: Handle User Input

* Use a loop to repeatedly display the menu and prompt the user for a choice.
* Call the appropriate function based on the user's selection.
* Ensure input validation for numeric values and string lengths.

## Step 5: Terminate Program

* Exit the loop when the user selects the Exit option.

# Code:

import java.util.Scanner;

class Employee { int id;

String name; String department; double salary;

Employee(int id, String name, String department, double salary) { this.id = id;

this.name = name; this.department = department; this.salary = salary;

}

void displayEmployee() {

System.*out*.printf("ID: %d, Name: %s, Department: %s, Salary: %.2f\n", id, name, department, salary);

}

}

public class EmployeeManagement { public static void main(String[] args) {

Scanner scanner = new Scanner(System.*in*);

System.*out*.print("Enter the number of employees: "); int n = scanner.nextInt();

scanner.nextLine(); // consume newline

Employee[] employees = new Employee[n]; int count = 0;

while (true) { System.*out*.println("\nMenu:"); System.*out*.println("1. Add Employee");

System.*out*.println("2. Display Employees"); System.*out*.println("3. Exit"); System.*out*.print("Choose an option: ");

int choice = scanner.nextInt(); scanner.nextLine(); // consume newline

switch (choice) { case 1:

if (count < n) {

System.*out*.print("Enter Employee ID: "); int id = scanner.nextInt(); scanner.nextLine(); // consume newline

System.*out*.print("Enter Employee Name: "); String name = scanner.nextLine();

System.*out*.print("Enter Employee Department: "); String department = scanner.nextLine();

System.*out*.print("Enter Employee Salary: "); double salary = scanner.nextDouble();

employees[count] = new Employee(id, name, department, salary); count++;

System.*out*.println("Employee added successfully!");

} else {

System.*out*.println("Employee array is full!");

}

break;

case 2:

if (count == 0) {

System.*out*.println("No employees to display.");

} else {

System.*out*.println("\nEmployee Details:"); for (int i = 0; i < count; i++) {

employees[i].displayEmployee();

}

}

break;

case 3:

System.*out*.println("Exiting program. Goodbye!"); scanner.close();

return;

default:

System.*out*.println("Invalid choice. Please try again.");

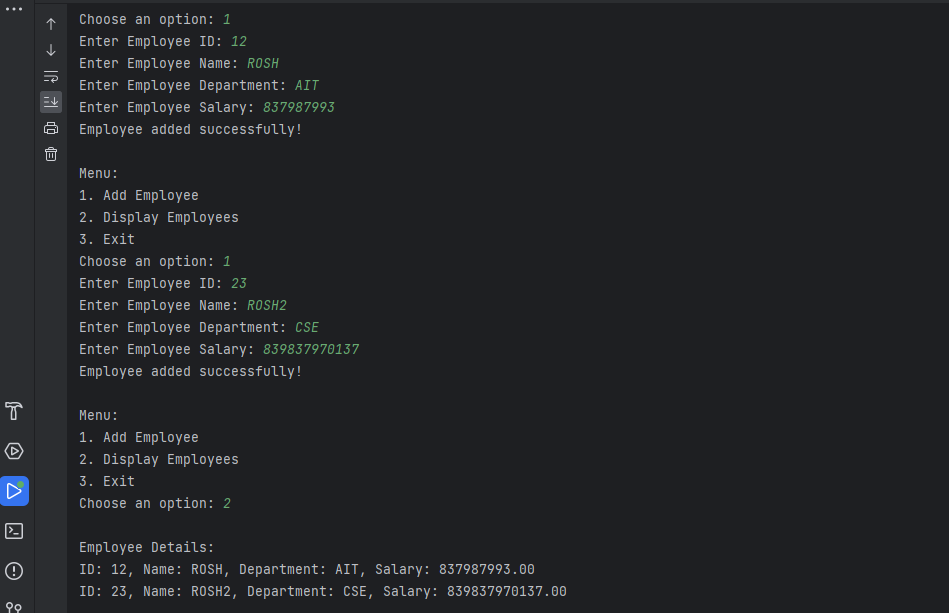
}

}

}

}

# Output:



**Learning Outcomes:**

1. Demonstrate: Apply key concepts to real-world scenarios to showcase understanding.
2. Analyze: Critically evaluate information, identify patterns, and draw meaningful conclusions.
3. Create: Develop original work, including presentations, reports, or projects, to exhibit comprehension and skills.
4. Communicate: Convey ideas and findings effectively through oral and written communication.
5. Collaborate: Contribute to group projects and exhibit strong teamwork capabilities in a collaborative environment.