**Exercise 4: Employee Management System**

\* Arrays in Memory:

- Arrays are stored in contiguous memory blocks.

- Indexing is done in O(1) time because the memory address of any element

can be computed using: base\_address + index \* element\_size.

\* Advantages:

- Fast access using index (O(1))

- Easy to implement

\* Limitations:

- Fixed size (static memory allocation)

- Insertion/deletion can be expensive (O(n))

- Not suitable for dynamic datasets

public class EmployeeManagementSystem {

static class Employee {

int employeeId;

String name;

String position;

double salary;

public Employee(int employeeId, String name, String position, double salary) {

this.employeeId = employeeId;

this.name = name;

this.position = position;

this.salary = salary;

}

@Override

public String toString() {

return "[" + employeeId + "] " + name + " - " + position + " ($" + salary + ")";

}

}

static class EmployeeArray {

private Employee[] employees;

private int size;

public EmployeeArray(int capacity) {

employees = new Employee[capacity];

size = 0;

}

public void addEmployee(Employee employee) {

if (size >= employees.length) {

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

return;

}

employees[size++] = employee;

}

// Traverse employees (Time: O(n))

public void listEmployees() {

if (size == 0) {

System.out.println("No employees available.");

return;

}

for (int i = 0; i < size; i++) {

System.out.println(employees[i]);

}

}

// Search by ID (Time: O(n))

public Employee searchEmployee(int employeeId) {

for (int i = 0; i < size; i++) {

if (employees[i].employeeId == employeeId) {

return employees[i];

}

}

return null;

}

// Delete by ID (Time: O(n))

public boolean deleteEmployee(int employeeId) {

for (int i = 0; i < size; i++) {

if (employees[i].employeeId == employeeId) {

// Shift elements left

for (int j = i; j < size - 1; j++) {

employees[j] = employees[j + 1];

}

employees[--size] = null;

return true;

}

}

return false;

}

}

public static void main(String[] args) {

EmployeeArray employeeArray = new EmployeeArray(5); // capacity = 5

employeeArray.addEmployee(new Employee(101, "Alice", "Manager", 80000));

employeeArray.addEmployee(new Employee(102, "Bob", "Developer", 60000));

employeeArray.addEmployee(new Employee(103, "Charlie", "Designer", 55000));

System.out.println("All Employees:");

employeeArray.listEmployees();

System.out.println("\nSearching for Employee ID 102:");

Employee emp = employeeArray.searchEmployee(102);

System.out.println(emp != null ? emp : "Employee not found");

System.out.println("\nDeleting Employee ID 101...");

boolean deleted = employeeArray.deleteEmployee(101);

System.out.println(deleted ? "Deleted successfully." : "Employee not found.");

System.out.println("\nAll Employees After Deletion:");

employeeArray.listEmployees();

}

}