**Collection Assignment**

**Consider the below class structure:**

1 to 1

1 to m

addressId

Addressline1 City

state

**Address**

employeeId

firstName lastName Salary

dateOfjoining Department Address

**Employee**

departmentId departmentName location

**Department**

Perform below operations with employee object. Create a menu driven approach to achieve the following:

1. Save Unique Employee
2. Sort Employee
   1. ByEmployeeId
   2. ByFirstName
   3. ByLastName
   4. BySalary
   5. ByAddress
   6. ByDeprtment
3. Exit

Create appropriate Class structure below validation:

* 1. Employee ID should be 5Digits and 2 chars (12345\_FS)
  2. Salary between 20000 to 5lak
  3. firstName, lastName should be alphabets.
  4. Join date must be today/future date.

**Program:**-

//Employee class Starts

**package** capgemini;

**import** java.util.\*;

**public** **class** Employee {

**private** String employeeID;

**private** String firstName, lastName;

**private** **float** salary;

**private** Date dateOfJoining;

**private** Department department;

**private** Address address;

**public** Employee(String employeeID, String firstName, String lastName, **float** salary, Date dateOfJoining,

Department department, Address address)

{

**super**();

**this**.employeeID = employeeID;

**this**.firstName = firstName;

**this**.lastName = lastName;

**this**.salary = salary;

**this**.dateOfJoining = dateOfJoining;

**this**.department = department;

**this**.address = address;

}

**public** String getEmployeeID() {

**return** employeeID;

}

**public** **void** setEmployeeID(String employeeID) {

**this**.employeeID = employeeID;

}

**public** String getFirstName() {

**return** firstName;

}

**public** **void** setFirstName(String firstName) {

**this**.firstName = firstName;

}

**public** String getLastName() {

**return** lastName;

}

**public** **void** setLastName(String lastName) {

**this**.lastName = lastName;

}

**public** **float** getSalary() {

**return** salary;

}

**public** **void** setSalary(**float** salary) {

**this**.salary = salary;

}

**public** Date getDateOfJoining() {

**return** dateOfJoining;

}

**public** **void** setDateOfJoining(Date dateOfJoining) {

**this**.dateOfJoining = dateOfJoining;

}

**public** Department getDepartment() {

**return** department;

}

**public** **void** setDepartment(Department department) {

**this**.department = department;

}

**public** Address getAddress() {

**return** address;

}

**public** **void** setAddress(Address address) {

**this**.address = address;

}

@Override

**public** String toString() {

**return** "[employeeID=" + employeeID + ",\n firstName=" + firstName + ",\n lastName=" + lastName

+ ",\n salary=" + salary + ", dateOfJoining=" + dateOfJoining + ",\n department=" + department.toString()

+ ",\n address=" + address.toString() + "]";

}

}

//Address class starts

**package** capgemini;

**public** **class** Address {

**private** String addressID;

**private** String addressLine1,city,state;

**public** Address(String addressID, String addressLine1, String city, String state)

{

**this**.addressID = addressID;

**this**.addressLine1 = addressLine1;

**this**.city = city;

**this**.state = state;

}

**public** String getAddressID() {

**return** addressID;

}

**public** **void** setAddressID(String addressID) {

**this**.addressID = addressID;

}

**public** String getAddressLine1() {

**return** addressLine1;

}

**public** **void** setAddressLine1(String addressLine1) {

**this**.addressLine1 = addressLine1;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getState() {

**return** state;

}

**public** **void** setState(String state) {

**this**.state = state;

}

@Override

**public** String toString() {

**return** "addressID=" + addressID + ",\n addressLine1=" + addressLine1 + ",\n city=" + city + ",\n state="

+ state;

}

}

//Department class Starts

**package** capgemini;

**public** **class** Department {

**private** String departmentId;

**private** String departmentName, location;

**public** Department(String departmentId,String departmentName,String location)

{

**this**.departmentId = departmentId;

**this**.departmentName = departmentName;

**this**.location = location;

}

**public** String getDepartmentId() {

**return** departmentId;

}

**public** **void** setDepartmentId(String departmentId) {

**this**.departmentId = departmentId;

}

**public** String getDepartmentName() {

**return** departmentName;

}

**public** **void** setDepartmentName(String departmentName) {

**this**.departmentName = departmentName;

}

**public** String getLocation() {

**return** location;

}

**public** **void** setLocation(String location) {

**this**.location = location;

}

@Override

**public** String toString() {

**return** "departmentId=" + departmentId + ",\n departmentName=" + departmentName + ",\n location="

+ location;

}

}

//CollectionAssignment class

**package** capgemini;

**import** java.util.\*;

**import** java.text.ParseException;

**import** java.text.SimpleDateFormat;

**public** **class** CollectionAssignment

{

**public** **static** ArrayList<Employee> populateEmp() **throws** ParseException

{

ArrayList<Employee> elist = **new** ArrayList<>();

Date doj=**null**;

doj = **new** SimpleDateFormat("dd-MM-yyyy").parse("29-06-2000");

elist.add(**new** Employee("10000\_FS","Luke","Wayne",200000,doj,**new** Department("254","IT","Noida"), **new** Address("111","line 01","Noida","UP")));

doj = **new** SimpleDateFormat("dd-MM-yyyy").parse("29-06-2000");

elist.add(**new** Employee("12000\_FS","Nell","Hartong",30000,doj,**new** Department("254","IT","Gurugram"), **new** Address("342","line 04","Gajraula","UP")));

doj = **new** SimpleDateFormat("dd-MM-yyyy").parse("01-04-1975");

elist.add(**new** Employee("10300\_JI","Hugh","Queen",25000,doj,**new** Department("876","Management","Pune"), **new** Address("343","line 01","Okhla","Delhi")));

doj = **new** SimpleDateFormat("dd-MM-yyyy").parse("11-12-1978");

elist.add(**new** Employee("19900\_SS","Olivia","Freecs",310000,doj,**new** Department("344","Service","Mumbai"), **new** Address("567","line 10","Rudrapur","Uttarakhand")));

doj = **new** SimpleDateFormat("dd-MM-yyyy").parse("01-01-1998");

elist.add(**new** Employee("09900\_HR","Theodora","Crain",45000,doj,**new** Department("654","HR","Bengalore"), **new** Address("899","line 08","Hasanpur","UP")));

**return** elist;

}

**public** **static** ArrayList<Employee> sortByEmpId(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getEmployeeID().compareTo(o2.getEmployeeID());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByFirstName(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getFirstName().compareTo(o2.getFirstName());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByLastName(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getLastName().compareTo(o2.getLastName());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortBySalary(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** ((Float)o1.getSalary()).compareTo(o2.getSalary());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByDepartmentId(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getDepartment().getDepartmentId().compareTo(o2.getDepartment().getDepartmentId());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByDepartmentName(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getDepartment().getDepartmentName().compareTo(o2.getDepartment().getDepartmentName());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByDepartmentLocation(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getDepartment().getLocation().compareTo(o2.getDepartment().getLocation());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByAddressId(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getAddress().getAddressID().compareTo(o2.getAddress().getAddressID());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByAddressLine1(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getAddress().getAddressLine1().compareTo(o2.getAddress().getAddressLine1());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByAddressCity(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getAddress().getCity().compareTo(o2.getAddress().getCity());

}

});

**return** emplist;

}

**public** **static** ArrayList<Employee> sortByAddressState(ArrayList<Employee> emplist)

{

Collections.*sort*(emplist, **new** Comparator<Employee>() {

@Override

**public** **int** compare(Employee o1, Employee o2)

{

**return** o1.getAddress().getState().compareTo(o2.getAddress().getState());

}

});

**return** emplist;

}

**public** **static** **void** main(String [] args) **throws** ParseException

{

ArrayList<Employee> employeeList = *populateEmp*();

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

**try**(Scanner sc = **new** Scanner(System.***in***)) {

System.***out***.println("Employees before sorting.");

System.***out***.println();

System.***out***.println("Sort Employee\r\n"

+ "a. ByEmployeeId\r\n"

+ "b. ByFirstName\r\n"

+ "c. ByLastName\r\n"

+ "d. BySalary\r\n"

+ "e. ByAddress\r\n"

+ "f. ByDeprtment");

**if** (sc.hasNext()) {

String choice = sc.next();

**switch**(choice) {

**case** "a":

employeeList = *sortByEmpId*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

System.***out***.println();

**break**;

**case** "b":

employeeList = *sortByFirstName*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

**break**;

**case** "c":

employeeList = *sortByLastName*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

**break**;

**case** "d":

employeeList = *sortBySalary*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

**break**;

**case** "e":

System.***out***.println("Choose address: \n1. By city\n2. By State\n3. By address\n4. By id");

**int** addressChoice = sc.nextInt();

**if** (addressChoice == 1) {

employeeList = *sortByAddressCity*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

} **else** **if** (addressChoice == 2) {

employeeList = *sortByAddressState*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

}**else** **if** (addressChoice == 3) {

employeeList = *sortByAddressLine1*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

}**else** **if** (addressChoice == 4) {

employeeList = *sortByAddressId*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

}

**break**;

**case** "f":

System.***out***.println("Choose how to want to sort.: \n1. By department id\n2. By department name\n3. By department location");

**int** deptChoice = sc.nextInt();

**if** (deptChoice == 1) {

employeeList = *sortByDepartmentId*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

} **else** **if** (deptChoice == 2) {

employeeList = *sortByDepartmentName*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

}**else** **if** (deptChoice == 3) {

employeeList = *sortByDepartmentLocation*(employeeList);

**for**(Employee emp : employeeList)

{

System.***out***.println(emp.toString());

}

}

**break**;

}

}

}

}

}

Output:-





