Program 7 : 60 points: File Input/Output Due: Thursday, November 19th, 11:59

**Objectives: The focus of this assignment is the use of both character sequential files and Object Serialized files**

**Program Description:**

The EmployeeManager will now have the ability to save the current Employees to a file, as well as load Employees from a previous session. Additionally, it will be capable of processing files that will apply multiple Employee updates.

A total of fourteen classes are required.

Queue – (From previous assignment)

ListNode – (From previous assignment)

LikedList – Implementation of the LinkedList data structure of Comparables

EmptyListException – (From previous assignment)

ArrayList (From previous assignment)

InvalidEmployeeNumberException (From previous assignment)

InvalidSizeException (From previous assignment)

MaximumCapacityException (From previous assignment)

Employee (Altered from previous assignment)

HourlyEmployee (From previous assignment)

SalaryEmployee (From previous assignment)

CommissionEmployee (From previous assignment)

EmployeeManager (Altered from previous assignment)

EmployeeDriver (From previous assignment)

Changes to Employee

The Employee class must now implement the Serializable interface in order to be used with Object Serialization

|  |
| --- |
| Employee {abstract} implements Comparable<Employee>, **Serializable** |
| - String firstName  - String lastName  - char middleInitial  - boolean fulltime  - char gender  - int employeeNum |
| <<constructor>> Employee (fn : String, ln : String, m : char, g : char, empNum : int, ft : boolean ) throws InvalidEmployeeNumberException  + getEmployeeNumber() : int  + setEmployeeNumber(empNum : int)  + getFirstName() : String  + getLastName() : String  + setFirstName(fn: String)  + setLastName(ln : String)  + setMiddleI(m : char)  + setGender(g : char)  + equals(e2 : Object) : Boolean  + toString() : String  + calculateWeeklyPay() : double {abstract}  + annualRaise() {abstract}  + holidayBonus() : double {abstract}  + resetWeek() {abstract}  + compareTo(Employee e) |

Changes to EmployeeManager

The EmployeeManager will now be able to save and load the Employees from session to session using Serialized files. It will also be able to process text files to quickly update multiple Employees.

|  |
| --- |
| EmployeeManager |
| - employees : ArrayList<Employee>  - employeeMax : final int = 10  - hourlyList : LinkedList<Employee>  - salaryList : LinkedList<Employee>  - commissionList : LinkedList<Employee>  - vacationRequests : Queue<Employee> |
| <<constructor>> EmployeeManager()  + addEmployee( type : int, fn : String, ln : String, m : char, g : char, en : int, ft : boolean, amount : double) throws InvalidEmployeeNumberException  + removeEmployee( index : int)  + listAll()  + listHourly()  + listSalary()  + listCommision()  + resetWeek()  + calculatePayout() : double  + getIndex( empNum : int ) : int  + annualRaises()  + holidayBonuses() : double  + increaseHours( index : int, amount : double)  + increaseSales( index : int, amount : double)  + findAllBySubstring(find : String) : Employee[]  - RabinKarp(name : String, find : String) : int  - stringHash(s : String) : int  - charNumericValue(c : char) : int  - RabinKarpHashes(s : String, hashes : int[], pos : int, length : int) : int  - linearSearchRecursive(nameHashes : int[], findHash : int, pos : int) : int  + sort()  + addRequest(empNum : int) : boolean  + viewNextRequest() : Employee  + grantNextRequest() : Employee  + outputRequests()  **+ loadEmployees(employeeFile : String, requestFile : String) boolean**  **+saveEmployees(employeeFile : String, requestFile : String) boolean**  **+processUpdates(fileName : String) boolean** |

public boolean loadEmployees(String employeeFile, String requestFile)

Given the name of a serialized file (employeeFile), this method attempts to read in Employees from a previous session. This method will return true if the load of Employees was successful and false if it was not successful (file doesn’t exists, ClassNotFoundException, or other IOException). Before reading in the Employees it will clear all of the lists to ensure it is loading into an empty set of Employees. This includes the ArrayList, the LinkedLists, and the Queue. If a MaximumCapacityException is encountered while adding Employees output “Maximum Capacity Reached”, and return true. The method will also open the text file (requestFile) to rebuild the vacation request Queue.

public boolean saveEmployees(String employeeFile, String requestFile)

This method will store all of the Employees as Serialized objects in the employeeFile and the Employee numbers in the correct order of those in the vacation Queue as plain text.

public boolean processUpdates(String fileName)

Given the name of a text file, this method will process the update requests within the file. A valid update request is in the form:

EMPLOYEE\_NUMBER AMOUNT

Where the two values are separated by a space. Employee numbers are ints, amount can be a double. It will skip updates with a negative amount. If an improperly formatted update request is reached it will skip that update **and proceed to the next update**.

To correctly apply an update you must call upon either increaseHours or increaseSales method of the EmployeeManager, depending on the type of employee that has the given Employee number. Employee numbers that do not exist or Employee numbers of SalaryEmployees are skipped.

Changes to EmployeeDriver

Before the Main Menu loop call upon the EmployeeManager’s loadEmployees with the file name of “employees.ser” and “requests.dat”. If the load is successful output “Employees Loaded”, if it was not successful output “Employees Not Loaded”.

Upon exit call upon the EmployeeManager’s saveEmployees. After the “Thank you for using the Employee Manager!” message, if the save was successful output “Employees stored”, if unsuccessful output “Employes not stored”.

Add a new menu option, “Employee Updates” after “Grant Request”. Employee Updates will now be option 13, moving Exit to option 14. This option will ask the user for the name of an update file, read it in and call upon the EmployeeManager’s processUpdates method. If this is done successfully output “Updates processed successfully”, otherwise output “Updates not processed”.

The first time you execute the loading of files will not succeed since there is no “employees.ser” file. After that it should always succeed.

If you need to retest your program from scratch, that is from the first run, you can remove the .ser file that is created with the rm command.

rm employees.ser

rm requests.dat

**Other Notes:**

* Import classes to files as necessary
* When loading Employees be sure to add them to their own lists as well!
* Name the file for the Serialized Employees “employees.ser” and the vacation request Employee numbers “requests.dat”

# **Submitting Your Program Files:**

The assignment is automatically collected from your account on the “Loki” machine on the due date. You must put your source code in the correct directory in your account in order for this to work. Your directory name includes the course, section, semester and the assignment number. This is case-sensitive; **all alphas are upper-case**. This directory must be created in your home directory.

Submission Directory:

**CSCI-1620-1-F15-A7**

**Example (user input underlined, bolded comments not in actual output)**

To begin with there is no “employees.ser” file. The contents for the update files used for this example are at the bottom

Employees Not Loaded **No files exists**

Main Menu

No Employees.

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Find Employee

9. Sort

10. View Vacation Requests

11. Add Vacation Request

12. Grant Vacation Request

13. Employee Updates

14. Quit

Enter Choice:

**After adding some Employees and exiting**

Main Menu

12348

Doe, John A.

Gender: M

Status: Full Time

Wage: 14.75

Hours Worked: 0.00

12345

Cavanaugh, Patrick B.

Gender: M

Status: Full Time

Wage: 10.00

Hours Worked: 0.00

12346

Bohning, Edith C.

Gender: F

Status: Part Time

Salary: 50000.00

12347

Pohnal, Rickie A.

Gender: F

Status: Full Time

Rate: 2.50

Sales: 0.00

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Find Employee

9. Sort

10. View Vacation Requests

11. Add Vacation Request

12. Grant Vacation Request

13. Employee Updates

14. Quit

Enter Choice: 14

Thank you for using the Employee Manager!

Employees stored **Employees successfully stored**

**Starting the driver again**

Employees Loaded **Employees from previous session loaded successfully**

Main Menu

12348

Doe, John A.

Gender: M

Status: Full Time

Wage: 14.75

Hours Worked: 0.00

12345

Cavanaugh, Patrick B.

Gender: M

Status: Full Time

Wage: 10.00

Hours Worked: 0.00

12346

Bohning, Edith C.

Gender: F

Status: Part Time

Salary: 50000.00

12347

Pohnal, Rickie A.

Gender: F

Status: Full Time

Rate: 2.50

Sales: 0.00

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Find Employee

9. Sort

10. View Vacation Requests

11. Add Vacation Request

12. Grant Vacation Request

13. Employee Updates

14. Quit

Enter Choice: 13 **Process update file**

Enter name of update file: update1

Updates processed successfully

Main Menu

12348

Doe, John A.

Gender: M

Status: Full Time

Wage: 14.75

Hours Worked: 35.75

12345

Cavanaugh, Patrick B.

Gender: M

Status: Full Time

Wage: 10.00

Hours Worked: 20.56

12346

Bohning, Edith C.

Gender: F

Status: Part Time

Salary: 50000.00

12347

Pohnal, Rickie A.

Gender: F

Status: Full Time

Rate: 2.50

Sales: 10000.54

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Find Employee

9. Sort

10. View Vacation Requests

11. Add Vacation Request

12. Grant Vacation Request

13. Employee Updates

14. Quit

Enter Choice: 13 **Process another**

Enter name of update file: update2

Updates processed successfully

Main Menu

12348 **Note the update with a negative value was not process, and the update**

Doe, John A. **after the incorrectly formatted update still occurred**

Gender: M

Status: Full Time

Wage: 14.75

Hours Worked: 35.75

12345

Cavanaugh, Patrick B.

Gender: M

Status: Full Time

Wage: 10.00

Hours Worked: 20.56

12346

Bohning, Edith C.

Gender: F

Status: Part Time

Salary: 50000.00

12347

Pohnal, Rickie A.

Gender: F

Status: Full Time

Rate: 2.50

Sales: 10500.54

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Find Employee

9. Sort

10. View Vacation Requests

11. Add Vacation Request

12. Grant Vacation Request

13. Employee Updates

14. Quit

Enter Choice: 14 **Exit**

Thank you for using the Employee Manager!

Employees stored

**Restarting one more time to show updates are also stored**

Employees Loaded

Main Menu

12348

Doe, John A.

Gender: M

Status: Full Time

Wage: 14.75

Hours Worked: 35.75

12345

Cavanaugh, Patrick B.

Gender: M

Status: Full Time

Wage: 10.00

Hours Worked: 20.56

12346

Bohning, Edith C.

Gender: F

Status: Part Time

Salary: 50000.00

12347

Pohnal, Rickie A.

Gender: F

Status: Full Time

Rate: 2.50

Sales: 10500.54

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Find Employee

9. Sort

10. View Vacation Requests

11. Add Vacation Request

12. Grant Vacation Request

13. Employee Updates

14. Quit

Enter Choice: 14

Thank you for using the Employee Manager!

Employees stored

**Contents of update files used for this example:**

update1

12348 35.75

12345 20.56

12346 90.12

12347 10000.54

update2

12345 -15.25

30.5 12348

12347 500

These are two sample updates. They do not encapsulate all possibilities and cases to check for. **Be sure to create and test your own update files.**