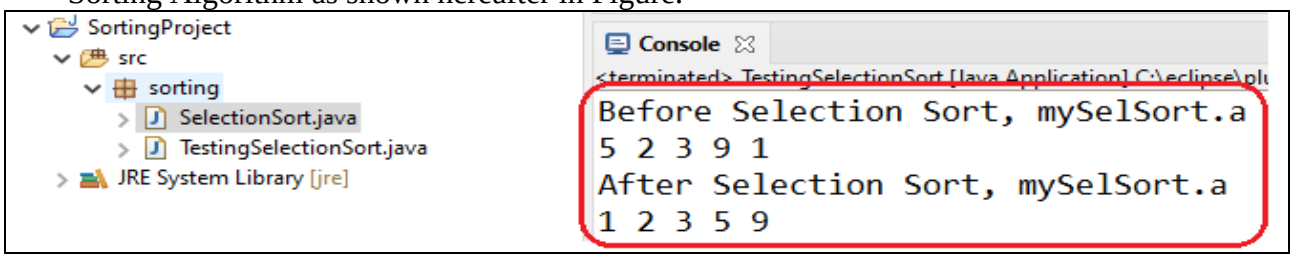
CEGEP VANIER COLLEGE
CENTRE FOR CONTINUING EDUCATION
Programming Algorithms and Patterns
420-930-VA

Teacher: Samir Chebbine **Lab 6** **Jul 16, 2024**

Lab 6: Using Design Patterns

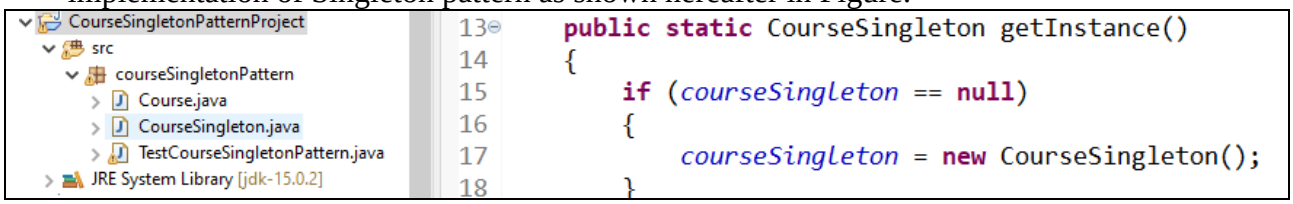
Complete all these following programs as explained during **classes.** All *missing coding statements* were provided there with explanation. **Create and Submit** a Word file ***Lab6ProgramminAlgorithmsandPatternsYourName.docx*** which includes **output screenshots** for every Java Project. Submit Java projects too.

1. **Sorting Algorithm**
   Create Selection Sort Java class as done during class to demonstrate the implementation of Sorting Algorithm as shown hereafter in Figure.

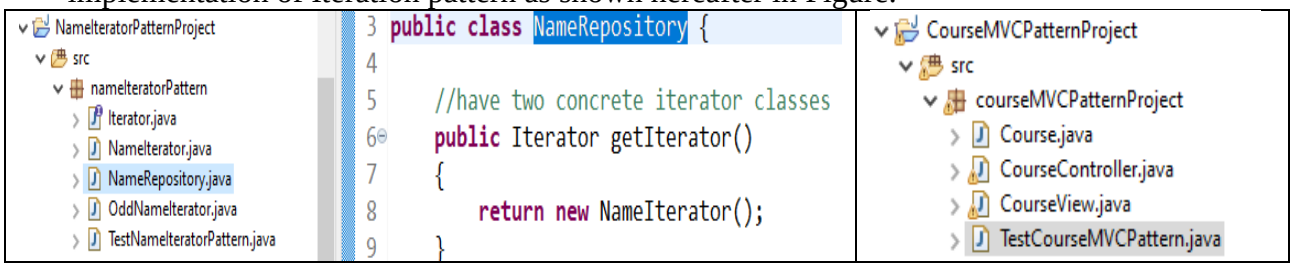


2. **Design Patterns:**

a) **Singleton Pattern**
   Create *CourseSingletonPatternProject* as done during class to demonstrate the implementation of Singleton pattern as shown hereafter in Figure.



b) **Iteration Pattern**
   Create *CourseSingletonPatternProject* as done during class to demonstrate the implementation of Iteration pattern as shown hereafter in Figure.



c) **Model View Controller (MVC) Pattern**
   Create *CourseMVCPatternProject* to demonstrate the implementation of MVC pattern classes (Course (M), CourseController(C), CourseView(V) classes) as shown in Figure.

3. Applying Lambda Expression to composed collection

You have been provided with input text file *Employee.in*.

Each line within *Employee.in* represents a record with following fields: *e_id* (Integer), *e_fname* (String), *e_lname* (String), *e_salary* (Double), *e_position* (String), *d_id* (Integer), *e_bonus* (of type ArrayList Double) containing data bonuses until number -999 is encountered (which is not part of data) and acting as sentinel (flag) at the end of each record.

a) Create a Java class *Employee*, to define data structure type, called *Employee*, which is designed to group data and methods into a single unit that *represents* a template of the fields (e_id, e_fname, e_lname, e_salary, e_d_id, e_position, and e_bonus of ArrayList of Double data type) used in creating the file *Employee*.

   a. Add Mutator **(**setter**)** methods and Accessor **(**getter**)** methods in `Employee` class.

b) Implement a Java Program "TestEmployeeProject" to store the records ***read from*** the input file *Employee.in* onto constructed ***HashMap*** where element is of `Employee` type.
Print all elements of the *HashMap keys applying Lambda expression* as shown hereafter.
Important: Your Java Program must run out of unlimited number of Employee records, and you need to take into account different number of bonuses for each employee until the number -999 is encountered.

Employee.in - Notepad

File Edit Format View Help

```
111     Amine   Kahn    265000  10      EX      500     632     430     -999
222     Roberts Sunny   35000   10      PT      99      5400    3000    4000    -999
123     Roberts Sandi   75000   30      SE      120     340     -999
433     McCall  Alex    66500   20      AD      1200    3700    -999
246     Houston Larry   150000  40      EX      450     233     1200    1640    -999
200     Shaw    Jinku   24500   30      PT      -999
135     Garner  Stanly  45000   20      PT      243     700     -999
543     Dev     Derek   80000   10      AD      365     950     840     -999
```

```
Print Employee Keys collection using Lambda Expression
111
222
123
433
246
200
135
543
```

c) Print then all elements of Employee *HashMap applying Lambda expression* invoking toString*( )* method as shown hereafter.

```
Print Employee info V collection using Lambda Expression
Employee [emp_no=111, fname=Amine, lname=Kahn, salary=265000.0, position=EX, bonus=[500.0, 632.0, 430.0]]
Employee [emp_no=222, fname=Roberts, lname=Sunny, salary=35000.0, position=PT, bonus=[99.0, 5400.0, 3000.0, 4000.0]]
Employee [emp_no=123, fname=Roberts, lname=Sandi, salary=75000.0, position=SE, bonus=[120.0, 340.0]]
Employee [emp_no=433, fname=McCall, lname=Alex, salary=66500.0, position=AD, bonus=[1200.0, 3700.0]]
Employee [emp_no=246, fname=Houston, lname=Larry, salary=150000.0, position=EX, bonus=[450.0, 233.0, 1200.0, 1640.0]]
Employee [emp_no=200, fname=Shaw, lname=Jinku, salary=24500.0, position=PT, bonus=[]]
Employee [emp_no=135, fname=Garner, lname=Stanly, salary=45000.0, position=PT, bonus=[243.0, 700.0]]
Employee [emp_no=543, fname=Dev, lname=Derek, salary=80000.0, position=AD, bonus=[365.0, 950.0, 840.0]]
```

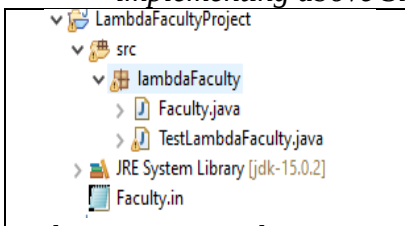
d) Print then all elements of Employee *HashMap applying Lambda expression on Employee HashMap* and printing its ArrayList bonuses and bonus taxes *applying Lambda expression* on ArrayList bonuses as shown hereafter. Set Prv_Tax =0.075, Fed_Tax=0.06

```
Print Faculty info V collection using Lambda Expression ver2
Employee [emp_no=111, fname=Amine, lname=Kahn, salary=265000.0, position=EX, bonus=[500.0, 632.0, 430.0]]
Bonus :500.0, Bonus Tax: 567.50
Bonus :632.0, Bonus Tax: 717.32
Bonus :430.0, Bonus Tax: 488.05
Employee [emp_no=222, fname=Roberts, lname=Sunny, salary=35000.0, position=PT, bonus=[99.0, 5400.0, 3000.0, 4000.0]]
Bonus :99.0, Bonus Tax: 112.37
Bonus :5400.0, Bonus Tax: 6129.00
Bonus :3000.0, Bonus Tax: 3405.00
Bonus :4000.0, Bonus Tax: 4540.00
Employee [emp_no=123, fname=Roberts, lname=Sandi, salary=75000.0, position=SE, bonus=[120.0, 340.0]]
Bonus :120.0, Bonus Tax: 136.20
Bonus :340.0, Bonus Tax: 385.90
Employee [emp_no=433, fname=McCall, lname=Alex, salary=66500.0, position=AD, bonus=[1200.0, 3700.0]]
Bonus :1200.0, Bonus Tax: 1362.00
Bonus :3700.0, Bonus Tax: 4199.50
Employee [emp_no=246, fname=Houston, lname=Larry, salary=150000.0, position=EX, bonus=[450.0, 233.0, 1200.0, 1640.0]]
Bonus :450.0, Bonus Tax: 510.75
Bonus :233.0, Bonus Tax: 264.46
Bonus :1200.0, Bonus Tax: 1362.00
Bonus :1640.0, Bonus Tax: 1861.40
Employee [emp_no=200, fname=Shaw, lname=Jinku, salary=24500.0, position=PT, bonus=[]]
Employee [emp_no=135, fname=Garner, lname=Stanly, salary=45000.0, position=PT, bonus=[243.0, 700.0]]
Bonus :243.0, Bonus Tax: 275.81
Bonus :700.0, Bonus Tax: 794.50
Employee [emp_no=543, fname=Dev, lname=Derek, salary=80000.0, position=AD, bonus=[365.0, 950.0, 840.0]]
Bonus :365.0, Bonus Tax: 414.28
Bonus :950.0, Bonus Tax: 1078.25
Bonus :840.0, Bonus Tax: 953.40
```

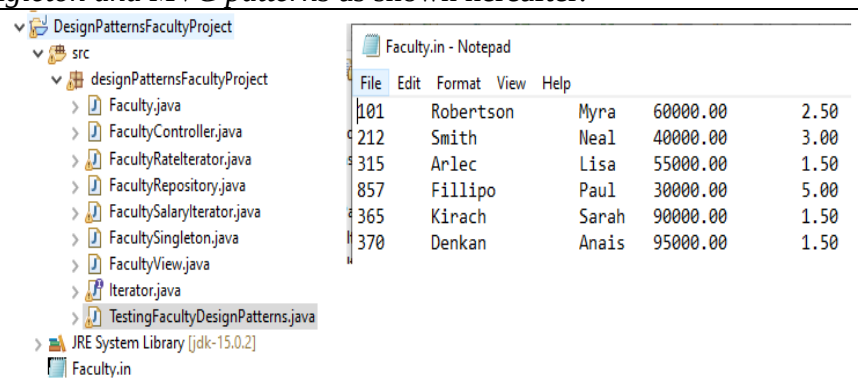
Lambda Expression on HashMap collection

Lambda Expression on ArrayList bonus collection

4. **Implementing Design Patterns (Singleton, Iteration and MVC) to HashMap collection Project**
   - Create *DesignPatternsFacultyProject* as shown in Figure, to store the records of the file *Faculty.in* (use delimiter \t to read *Faculty.in*) onto *HashMap* implementing following ***design patterns Singleton, Iteration and MVC patterns***.
   - Create a Model Java class *Faculty (the same as Lab 5)*, to define data structure type, called *Faculty*, which includes the following members:
     a. The private data members: *f_Id* (Integer), *f_Lname* (String), *f_Fname* (String), *f_Salary* (double), *f_BonusRate* (double). This order represents the columns in the file *Faculty.in*
     b. method *doCalc_Bonus() and doBonus_tax()* / Assuming f_tax = 0.075, p_tax=0.06
   - Implement *Singleton pattern* so that *testing code client* uses *only* one *single* object accessing database source (in this case *Faculty.in* input file*)* and manipulating *only one single HashMap collection to store every record of input file* into *HashMap* collection.
   - Implement *MVC pattern* for decoupling user-interface (view to display keys and values of HashMap Faculty collection using Lambda expression), Faculty data (model), and application logic (controller) used in *testing code client* to achieve separation of concerns.
   - Print all elements of the *HashMap keys* and all elements of faculty *HashMap values implementing above Singleton and MVC patterns* as shown hereafter.



Lab 5 project without using Design Patterns



Lab 6 project Implementing Design Patterns
Notice more classes were added but to increase Reusability and ease Extensibility and Maintainability

```
Implement Singleton and Use MVC patterns to Print Faculty Keys collection using Lambda Expression
370
212
101
857
315
365
```

FacultySingleton used to reference one HashMap Faculty collection.
View V within MVC to display the Faculty HashMap content.
Faculty View 1

```
Implement Singleton and Use MVC patterns to Print Faculty info V collection using Lambda Expression
Faculty [f_id=370, f_Lname=Denkan, f_Fname=Anais, f_salary=95000.0, f_bonusRate=1.5%,
 Faculty Bonus=1425.00$, Faculty Tax Bonus  =192.38$]
Faculty [f_id=212, f_Lname=Smith, f_Fname=Neal, f_salary=40000.0, f_bonusRate=3.0%,
 Faculty Bonus=1200.00$, Faculty Tax Bonus  =162.00$]
Faculty [f_id=101, f_Lname=Robertson, f_Fname=Myra, f_salary=60000.0, f_bonusRate=2.5%,
 Faculty Bonus=1500.00$, Faculty Tax Bonus  =202.50$]
Faculty [f_id=857, f_Lname=Fillipo, f_Fname=Paul, f_salary=30000.0, f_bonusRate=5.0%,
 Faculty Bonus=1500.00$, Faculty Tax Bonus  =202.50$]
Faculty [f_id=315, f_Lname=Arlec, f_Fname=Lisa, f_salary=55000.0, f_bonusRate=1.5%,
 Faculty Bonus=825.00$, Faculty Tax Bonus  =111.38$]
Faculty [f_id=365, f_Lname=Kirach, f_Fname=Sarah, f_salary=90000.0, f_bonusRate=1.5%,
 Faculty Bonus=1350.00$, Faculty Tax Bonus  =182.25$]
```

   - Update the previous data model faculty *HashMap* to new model faculty *HashMap sorted* with respect *to faculty bonus* invoking *doCalc_Bonus()* in order to print *updated View* implemented within MVC as shown hereafter.

```
--- Creating new values HashMap from Map collection (Sorted by Value doCalc_Bonus) ---

Implement Singleton and Use MVC patterns to Print Faculty Keys collection using Lambda Expression
315
212
365
370
101
857

Implement Singleton and Use MVC patterns to Print Faculty info V collection using Lambda Expression
Faculty [f_id=315, f_Lname=Arlec, f_Fname=Lisa, f_salary=55000.0, f_bonusRate=1.5%,
 Faculty Bonus=825.00$, Faculty Tax Bonus  =111.38$]
Faculty [f_id=212, f_Lname=Smith, f_Fname=Neal, f_salary=40000.0, f_bonusRate=3.0%,
 Faculty Bonus=1200.00$, Faculty Tax Bonus  =162.00$]
Faculty [f_id=365, f_Lname=Kirach, f_Fname=Sarah, f_salary=90000.0, f_bonusRate=1.5%,
 Faculty Bonus=1350.00$, Faculty Tax Bonus  =182.25$]
Faculty [f_id=370, f_Lname=Denkan, f_Fname=Anais, f_salary=95000.0, f_bonusRate=1.5%,
 Faculty Bonus=1425.00$, Faculty Tax Bonus  =192.38$]
Faculty [f_id=101, f_Lname=Robertson, f_Fname=Myra, f_salary=60000.0, f_bonusRate=2.5%,
 Faculty Bonus=1500.00$, Faculty Tax Bonus  =202.50$]
Faculty [f_id=857, f_Lname=Fillipo, f_Fname=Paul, f_salary=30000.0, f_bonusRate=5.0%,
 Faculty Bonus=1500.00$, Faculty Tax Bonus  =202.50$]
```

Using FacultyController to update the FacultyView displaying sorted Faculty HashMap content with respect to doCalc_Bonus()

- Implement *Iterator pattern* defining Iterator interface so that testing *code client* accesses *structured object representation (FacultyRepository)* without having to know the *internal data structure* (in this case Faculty HashMap collection) and define *two traversal protocols* for the used Faculty HashMap collection.
- Implement *FacultyRateIterator* class that defines a traversal iterator (two methods *hasNext() and next()*) of Faculty HashMap collection traversal with respect to a given Faculty bonus rate console input.
- Implement *FacultySalaryIterator* class that defines a traversal iterator (two methods *hasNext() and next()*) of Faculty HashMap collection sorted in descending order with respect to *value of faculty salary*.
- Skip through *FacultyRateIterator* object (via *FacultyRepository* object) of *Iteration pattern* to print all *iterated* faculty *HashMap values* and using *implemented above Singleton and MVC patterns* as shown hereafter.

```
Simple Faculty Traversal of Collection Using implemented Iterator Pattern (FacultyRateIterator)
and printing using View of MVC Pattern
Using filter() to skip through new Map Iterator that matches of input Faculty bonus rate in HashMap
Please enter Faculty bonus rate to Iterate its collection: 1.5
Faculty Info:
Faculty f_id=370
f_Lname=Denkan
f_Fname=Anais
f_salary=95000.0
f_bonusRate=1.5
Faculty Bonus=1425.00$
Faculty Tax Bonus  =192.38$

Faculty Info:
Faculty f_id=315
f_Lname=Arlec
f_Fname=Lisa
f_salary=55000.0
f_bonusRate=1.5
Faculty Bonus=825.00$
Faculty Tax Bonus  =111.38$

Faculty Info:
Faculty f_id=365
f_Lname=Kirach
f_Fname=Sarah
f_salary=90000.0
f_bonusRate=1.5
Faculty Bonus=1350.00$
Faculty Tax Bonus  =182.25$
```

Skip through *FacultyRateIterator* object (via *FacultyRepository* object) of *Iteration pattern* to print all elements of faculty *HashMap values using implemented above Singleton and MVC patterns*

Notice new View displaying Single Faculty Object within FacultyView (V) of MVC pattern

Faculty View 2

- Skip through *FacultySalaryIterator* object (via *FacultyRepository* object) of *Iteration pattern* to print all *iterated* faculty *HashMap values* and using *implemented above Singleton and MVC patterns* as shown hereafter.

```
Salary Faculty Traversal of Collection Using implemented Iterator Pattern (FacultySalaryIterator)
and printing using View of MVC Pattern
Skip through new Map Iterator in Faculty Salary Descending Order in HashMap
Faculty [f_id=370, f_Lname=Denkan, f_Fname=Anais, f_salary=95000.0, f_bonusRate=1.5%,
 Faculty Bonus=1425.00$, Faculty Tax Bonus  =192.38$]
Faculty [f_id=365, f_Lname=Kirach, f_Fname=Sarah, f_salary=90000.0, f_bonusRate=1.5%,
 Faculty Bonus=1350.00$, Faculty Tax Bonus  =182.25$]
Faculty [f_id=101, f_Lname=Robertson, f_Fname=Myra, f_salary=60000.0, f_bonusRate=2.5%,
 Faculty Bonus=1500.00$, Faculty Tax Bonus  =202.50$]
Faculty [f_id=315, f_Lname=Arlec, f_Fname=Lisa, f_salary=55000.0, f_bonusRate=1.5%,
 Faculty Bonus=825.00$, Faculty Tax Bonus  =111.38$]
Faculty [f_id=212, f_Lname=Smith, f_Fname=Neal, f_salary=40000.0, f_bonusRate=3.0%,
 Faculty Bonus=1200.00$, Faculty Tax Bonus  =162.00$]
Faculty [f_id=857, f_Lname=Fillipo, f_Fname=Paul, f_salary=30000.0, f_bonusRate=5.0%,
 Faculty Bonus=1500.00$, Faculty Tax Bonus  =202.50$]
```

Faculty View 3