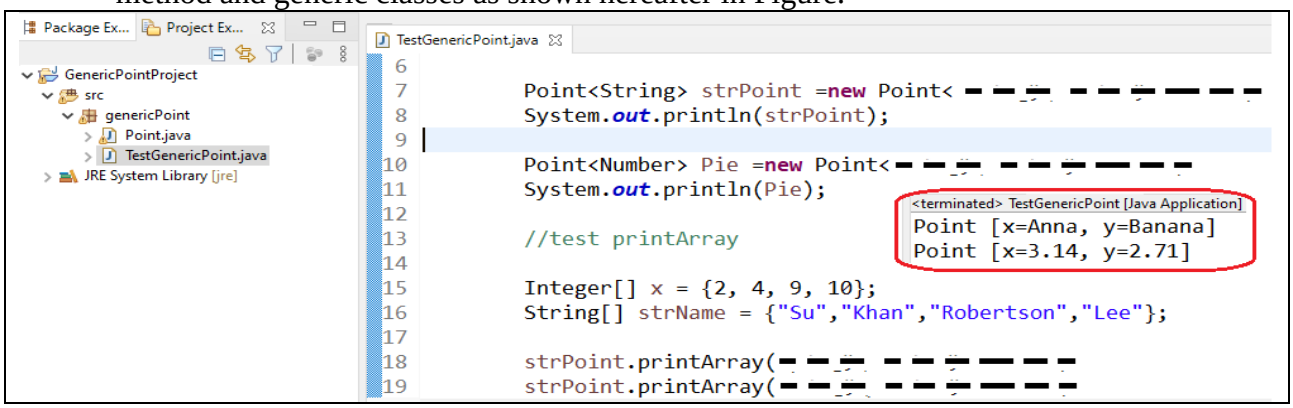
CEGEP VANIER COLLEGE
CENTRE FOR CONTINUING EDUCATION
Programming Algorithms and Patterns
420-930-VA

Teacher: Samir Chebbine **Lab 4** **Jun 25, 2024**

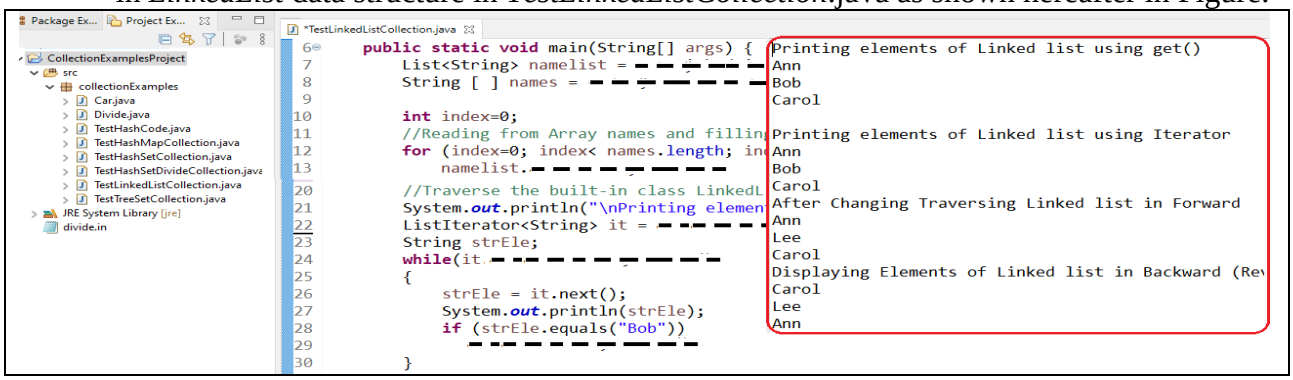
Lab 4: Generic Classes and Java Collections Framework

Complete all these following programs as explained during **classes.** All *missing coding statements* were provided there with explanation. **Create and Submit** a Word file ***Lab4ProgramminAlgorithmsandPatternsYourName.docx*** which includes **output screenshots** for every Java Project. Submit the Java projects too.

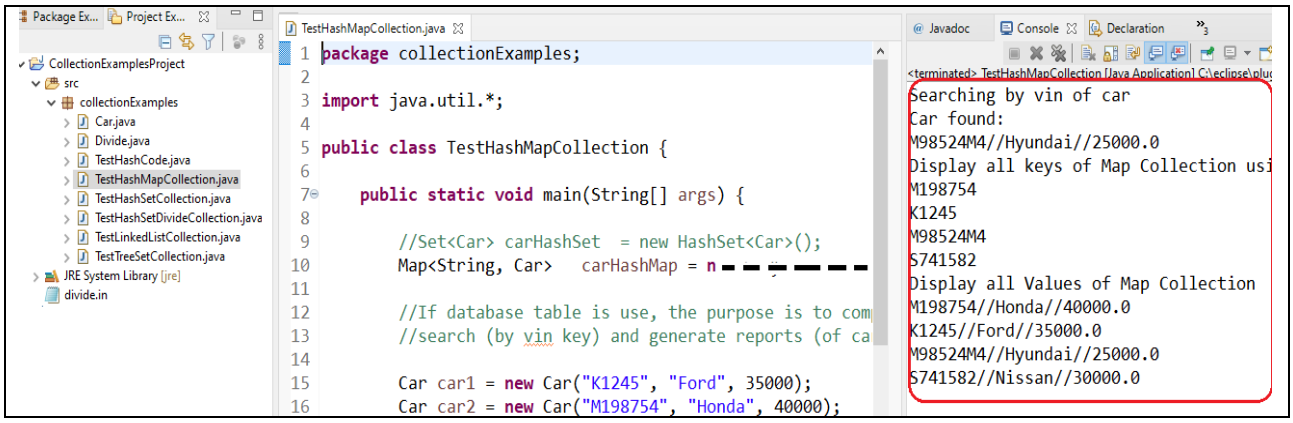
1. **Generic methods & Generic classes**
   a) Create *GenericPointProject* using Eclipse IDE for demonstrating the use of generic method and generic classes as shown hereafter in Figure.



2. **Java Collection Framework (LinkedList, ArrayList)**
   a) ListIterator interface
   Create a Java Project named *CollectionExamplesProject* to demonstrate the use of ListIterator methods (*hasNext(), next(), hasPrevious(), previous()*) when traversing built-in *LinkedList* data structure in *TestLinkedListCollection*.java as shown hereafter in Figure.



   b) TreeSet, HashSet, LinkedHashSet, TreeMap, HashMap, LinkedHashMap
   Create testing Java classes as done during Zoom class to demonstrate the use of *TreeSet, HashSet, LinkedHashSet, TreeMap, HashMap, LinkedHashMap* concrete classes as shown hereafter in Figure. Upload *CourseArrayListCollectionProject.*



3. Using ArrayList class and its methods add(), size(), get()

- Create *BookArrayListCollectionProject* as shown in Figure 1, to store records of the file *Book.in* (use delimiter \t to read *Book.in*) onto an *ArrayList* of Book class type using the method `add()`.
- Use class Book (from Lab3) to represent a single record (`b_id ,b_author,b_title,b_isbn, b_type, b_price`).
- Display number of elements of the *ArrayList* using the method `size()`.
- Print all elements of the *ArrayList* using the method `get()`.
- Print all elements of *ArrayList* using the method `next()` of `ListIterator` interface.
- Print all elements of *ArrayList* in reverse order using the method `previous()` of `ListIterator` interface.
- Add a record (13,"Joshawa Pierre","Python","1209845","BG",99.99) into *ArrayList* at index 2 using the method `add(int index, Book wrecord)`.

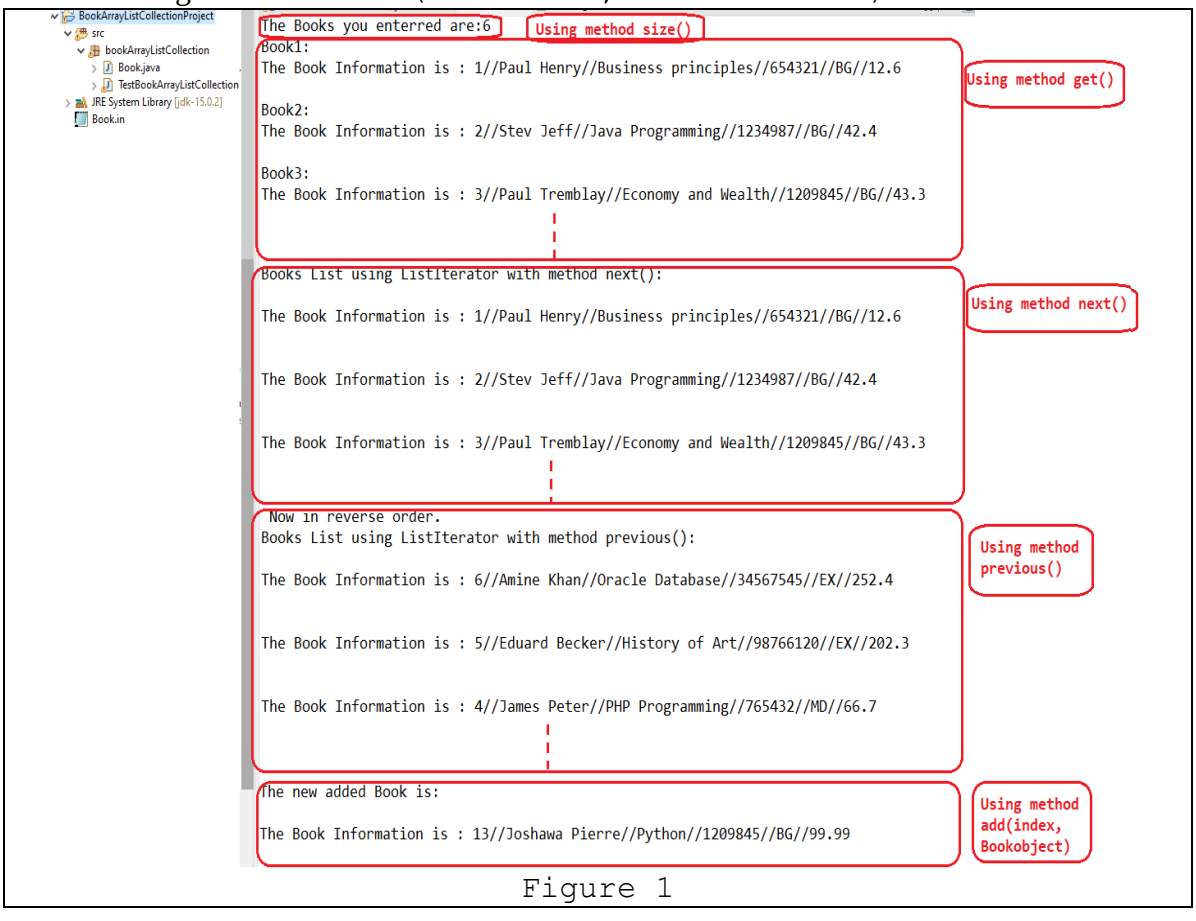


Figure 1

4. Using HashSet class and its methods add(), size()

- Create *CollectionTripProject* as shown in Figure 2, to store the records of the file *Trip.in* (use delimiter \t to read *Trip.in*) onto an *HashSet* using the method `add()`.
- Create a Java class *Trip*, to define data structure type, called `Trip`, which includes the following members:
  a. The private data members: *emp_id* (Integer), *emp_name* (String), *emp_address* (String), *emp_gasprice* (double), *emp_distance* (int), *emp_costhotel* (double), and *emp_costfood* (double). This order represents the columns in the file *Trip.in*
  b. Add Mutator **(**setter**)** methods in `Client` class to *modify* the values of private members.
  c. Add Accessor **(**getter**)** methods in `Client` class to *access* the values of private members.
  d. Add a method (*CalculateCostTrip()* that calculates, and returns the cost of a trip (cost trip = (*emp_distance* * *emp_gasprice*) + *emp_costhotel* + *emp_costfood*)
- Add every record stored as an object into *HashSet* using the method `add(Trip wrecord)`
- Display the number of elements of the *HashSet* using the method `size()`.
- Print all elements of the *HashSet.*
- Print all elements of the *HashSet* using the method `next()` of `Iterator` interface.

Figure 2

- Add a record (2,"Amine Khan", "Paris France", 1.11, 50, 75.00, 50.00) into the *HashSet* using the method `add(Trip wrecord)`.
- Explain *why* the record was added to the set despite that the set does not accept *duplicated* values as shown in Figure 2.
- Add statements to prevent end user to enter duplicated information related to Trip Employee objects when the *name of employee* is already in the set as shown in Figure 2.
- Add statements to store every record stored as an object into *LinkedHashSet* in order to display the information in the *same order* as found in the input file as shown in Figure 3.

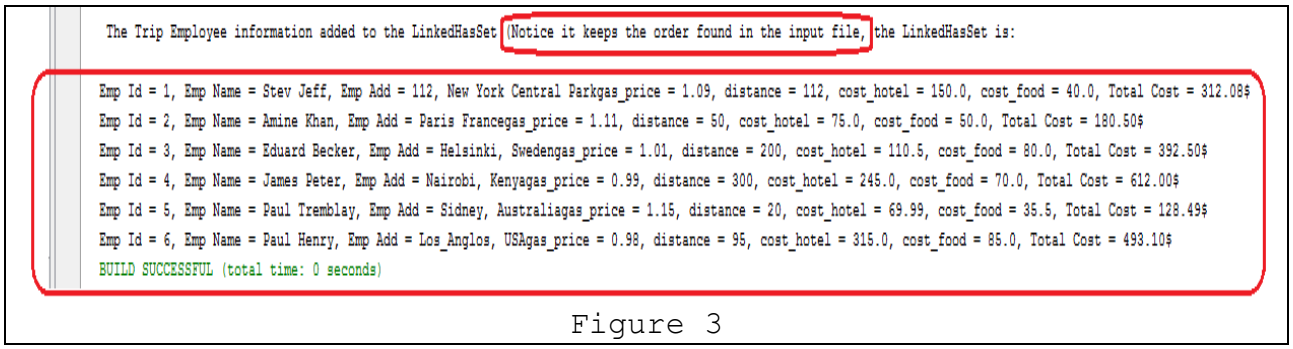


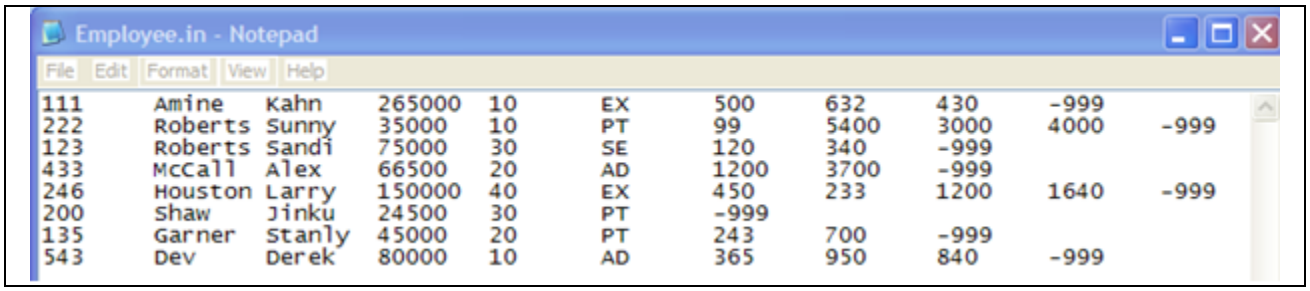
Figure 3

3. Application: Linked hash map and Linked List

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_bonus1* (Double), *e_bonus2* (Double), …, and so on until the 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, e_bonus) used in creating the file *Employee*.
   a. Add Mutator **(**setter**)** methods and Accessor **(**getter**)** methods in `Employee` class.

b) Implement a Java Program "TestEmployeeProject1" to display every employee in *Employee.in* and its corresponding employee bonuses while *you skip through input file* as in Figure 4.
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.



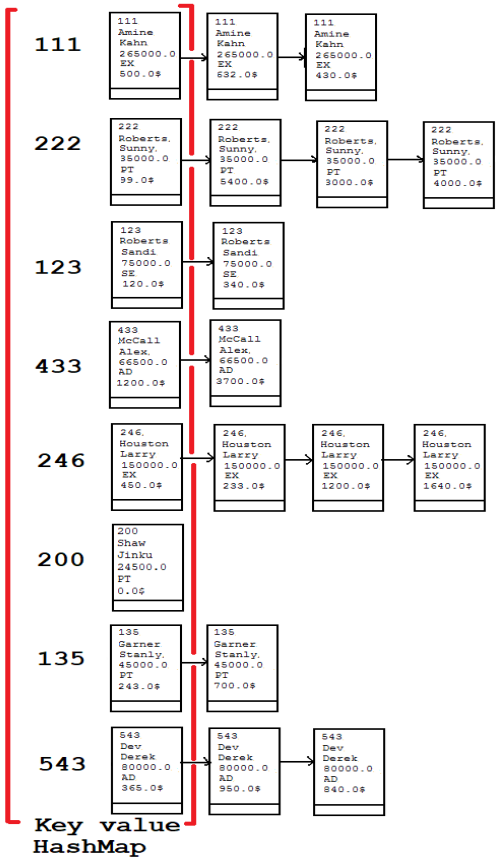
c) Implement a Java Program "TestEmployeeProject2" to store the records ***read from*** the input file *Employee.in* onto constructed ***HashMap*** and parallel ***linked list*** collection where each linked list node is of `Employee` type as shown hereafter.

You will get to construct ***eight linked list*** after reading records from (*Employee.in*).

Save ***the head of every linked*** list into a hash map *HashMap <key, value> collection* where key represents the *emp_id* and value represents the record of Employee class type.

You need to skip through the constructed data structure and display the output of Figure 5.

Add more Java statements TestEmployeeProject2 to search for a given emp_id within constructed HashMap /Parallel Linked List collection and display its bonuses and the sum of its bonuses as shown in Figure 6.




```
Employee N: 111, Employee Name: Amine Kahn, Employee Salary: 265000.00
    Bonus 1 : 500.00
    Bonus 2 : 632.00
    Bonus 3 : 430.00

Employee N: 222, Employee Name: Roberts Sunny, Employee Salary: 35000.00
    Bonus 1 : 99.00
    Bonus 2 : 5400.00
    Bonus 3 : 3000.00
    Bonus 4 : 4000.00

Employee N: 123, Employee Name: Roberts Sandi, Employee Salary: 75000.00
    Bonus 1 : 120.00
    Bonus 2 : 340.00

Employee N: 433, Employee Name: McCall Alex, Employee Salary: 66500.00
    Bonus 1 : 1200.00
    Bonus 2 : 3700.00

Employee N: 246, Employee Name: Houston Larry, Employee Salary: 150000.00
    Bonus 1 : 450.00
    Bonus 2 : 233.00
    Bonus 3 : 1200.00
    Bonus 4 : 1640.00

Employee N: 200, Employee Name: Shaw Jinku, Employee Salary: 24500.00

Employee N: 135, Employee Name: Garner Stanly, Employee Salary: 45000.00
    Bonus 1 : 243.00
    Bonus 2 : 700.00

Employee N: 543, Employee Name: Dev Derek, Employee Salary: 80000.00
    Bonus 1 : 365.00
    Bonus 2 : 950.00
    Bonus 3 : 840.00
```

Figure 4

```
Display all Values of Map Collection

Displaying the components of the linked list for emp_id: 111
The Employee Information is :111//Amine//Kahn//265000.00//EX//500.00$
The Employee Information is :111//Amine//Kahn//265000.00//EX//632.00$
The Employee Information is :111//Amine//Kahn//265000.00//EX//430.00$
Total bonus for emp_id: 111 is 1562.00$

Displaying the components of the linked list for emp_id: 222
The Employee Information is :222//Roberts//Sunny//35000.00//PT//99.00$
The Employee Information is :222//Roberts//Sunny//35000.00//PT//5400.00$
The Employee Information is :222//Roberts//Sunny//35000.00//PT//3000.00$
The Employee Information is :222//Roberts//Sunny//35000.00//PT//4000.00$
Total bonus for emp_id: 222 is 12499.00$

Displaying the components of the linked list for emp_id: 123
The Employee Information is :123//Roberts//Sandi//75000.00//SE//120.00$
The Employee Information is :123//Roberts//Sandi//75000.00//SE//340.00$
Total bonus for emp_id: 123 is 460.00$

Displaying the components of the linked list for emp_id: 433
The Employee Information is :433//McCall//Alex//66500.00//AD//1200.00$
The Employee Information is :433//McCall//Alex//66500.00//AD//3700.00$
Total bonus for emp_id: 433 is 4900.00$

Displaying the components of the linked list for emp_id: 246
The Employee Information is :246//Houston//Larry//150000.00//EX//450.00$
The Employee Information is :246//Houston//Larry//150000.00//EX//233.00$
The Employee Information is :246//Houston//Larry//150000.00//EX//1200.00$
The Employee Information is :246//Houston//Larry//150000.00//EX//1640.00$
Total bonus for emp_id: 246 is 3523.00$

Displaying the components of the linked list for emp_id: 200
The Employee Information is :200//Shaw//Jinku//24500.00//PT//0.00$
Total bonus for emp_id: 200 is 0.00$

Displaying the components of the linked list for emp_id: 135
The Employee Information is :135//Garner//Stanly//45000.00//PT//243.00$
The Employee Information is :135//Garner//Stanly//45000.00//PT//700.00$
Total bonus for emp_id: 135 is 943.00$

Displaying the components of the linked list for emp_id: 543
The Employee Information is :543//Dev//Derek//80000.00//AD//365.00$
The Employee Information is :543//Dev//Derek//80000.00//AD//950.00$
The Employee Information is :543//Dev//Derek//80000.00//AD//840.00$
Total bonus for emp_id: 543 is 2155.00$
```

Figure 5

```
Please enter emp id for search: 222
Employee foundThe Employee Information is :222//Roberts//Sunny//35000.00//PT//99.00$

Displaying the components of the linked list for emp_id: 222
The Employee Information is :222//Roberts//Sunny//35000.00//PT//99.00$
The Employee Information is :222//Roberts//Sunny//35000.00//PT//5400.00$
The Employee Information is :222//Roberts//Sunny//35000.00//PT//3000.00$
The Employee Information is :222//Roberts//Sunny//35000.00//PT//4000.00$
Total bonus for emp_id: 222 is 12499.00$
```

Figure 6