Basics of Java

This document details the requirement specifications for the above-named project. Reach out to your SME / Trainer for any query.

Technology	Java
Document Type	Basics of Java Practice Exercise – Day 7
Author	MLA
Current Version	1.0
Current Version Date	16-07-2024
Status	Active

Document Control

Version	Change Date	Change Description	Changed By
1.0	16-07-2024	Document Creation	Vanitha G

Problem Statement 1: Implement the flexible data structures using Collection.

Solve following sub problems,

1. Write a program to add list of student names to ArrayList and it should find a particular name whether it exists or not in the list.
2. Create a Product class with Product Id & Product Name. Write a program to accept information of 10 products and store that in HashSet. Do following operations,
   a. Search a particular product in the HashSet.
   b. Remove a particular product from the HashSet by using product id.
   c. (Refer below table for the product list)

Product Id	Product Name
P001	Maruti 800
P002	Maruti Zen
P003	Maruti Dezire
P004	Maruti Alto

3. Implement LinkedList class for this problem
   a. Create an Employee class which will have details like EmployeeNo, EmployeeName and Address. You should pass value for EmployeeNo, EmployeeName and Address through constructor.
   b. Create a method addInput( ) which will add employee details to LinkedList.
   c. Create method display( ) which should display all data from LinkedList using forward and reverse order using Iterator and ListIterator interfaces.
   d. Note: addInput( ) and display( ) should not be member functions of Employee class.
4. Create a Phone Book having user interface like,
   a. Add new phone book entry
   b. Search Phone Number
   c. Quit

   Option i :it allows add name and Phone no.

   Option ii: it must take name as input from the user and based on that it should return phone No.

   Option iii: will terminate the program.

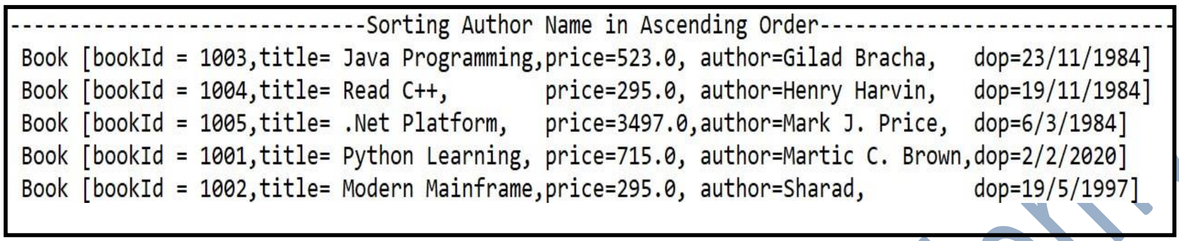
   Note: Use HashMap to store phone book entries.

5. Create a Book class with bookId, tile, price, date of publication and author. Override all the required methods such as toString, hashcode, equals, and compareTo. Implement natural ordering.

   Write a program that accepts information of 5 Book details and stores it in TreeSet.Do

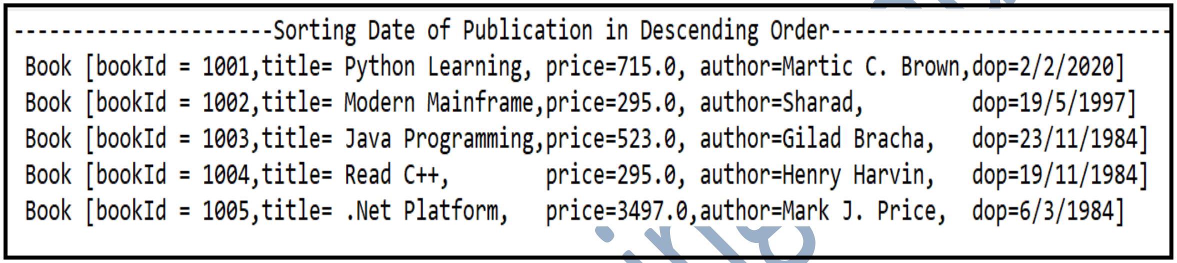
   following operations,

   a. Print all the Book details by sorting the author names in ascending order using Comparable.



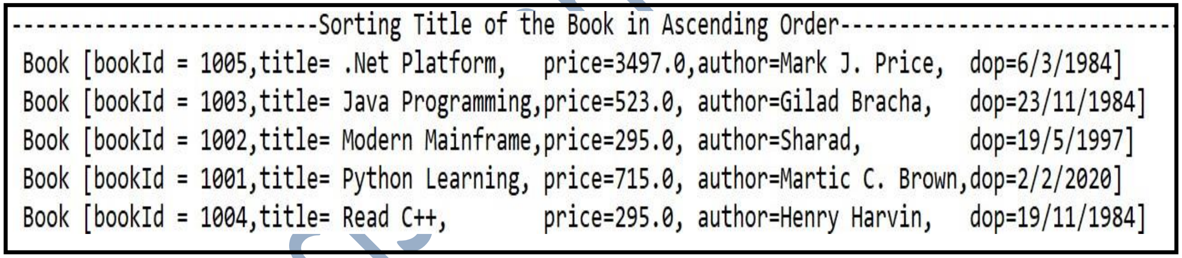
```
-----------------------------Sorting Author Name in Ascending Order----------------------------
 Book [bookId = 1003,title= Java Programming,price=523.0, author=Gilad Bracha,   dop=23/11/1984]
 Book [bookId = 1004,title= Read C++,        price=295.0, author=Henry Harvin,   dop=19/11/1984]
 Book [bookId = 1005,title= .Net Platform,   price=3497.0,author=Mark J. Price,  dop=6/3/1984]
 Book [bookId = 1001,title= Python Learning, price=715.0, author=Martic C. Brown,dop=2/2/2020]
 Book [bookId = 1002,title= Modern Mainframe,price=295.0, author=Sharad,         dop=19/5/1997]
```

   b. Print all the Book details by sorting the date of publication in descending order using Comparator.



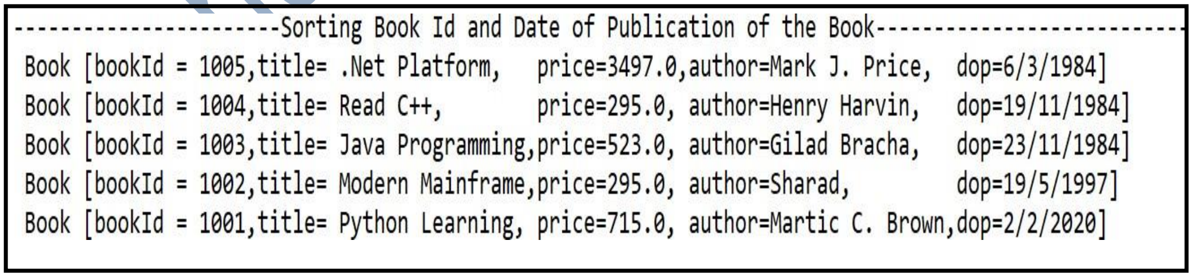
```
---------------------Sorting Date of Publication in Descending Order---------------------------
 Book [bookId = 1001,title= Python Learning, price=715.0, author=Martic C. Brown,dop=2/2/2020]
 Book [bookId = 1002,title= Modern Mainframe,price=295.0, author=Sharad,         dop=19/5/1997]
 Book [bookId = 1003,title= Java Programming,price=523.0, author=Gilad Bracha,   dop=23/11/1984]
 Book [bookId = 1004,title= Read C++,        price=295.0, author=Henry Harvin,   dop=19/11/1984]
 Book [bookId = 1005,title= .Net Platform,   price=3497.0,author=Mark J. Price,  dop=6/3/1984]
```

   c. Print all the Book details by sorting the title of the book in ascending order using Comparator.



```
-------------------------Sorting Title of the Book in Ascending Order---------------------------
 Book [bookId = 1005,title= .Net Platform,   price=3497.0,author=Mark J. Price,  dop=6/3/1984]
 Book [bookId = 1003,title= Java Programming,price=523.0, author=Gilad Bracha,   dop=23/11/1984]
 Book [bookId = 1002,title= Modern Mainframe,price=295.0, author=Sharad,         dop=19/5/1997]
 Book [bookId = 1001,title= Python Learning, price=715.0, author=Martic C. Brown,dop=2/2/2020]
 Book [bookId = 1004,title= Read C++,        price=295.0, author=Henry Harvin,   dop=19/11/1984]
```

   d. Print all the Book details by sorting the bookid in descending order and date of publicationin ascending order using Comparator.



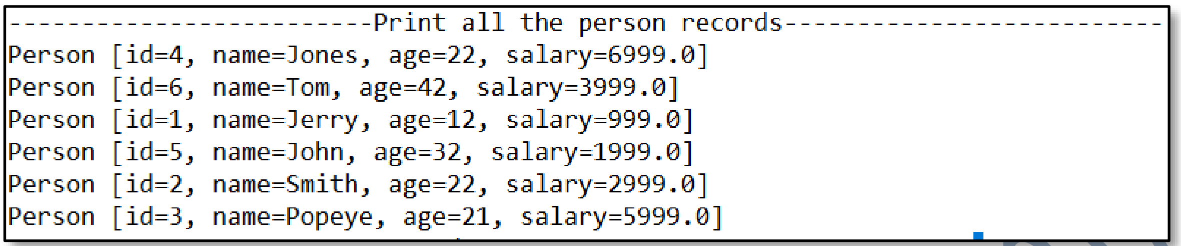
```
---------------------Sorting Book Id and Date of Publication of the Book-------------------------
 Book [bookId = 1005,title= .Net Platform,   price=3497.0,author=Mark J. Price,  dop=6/3/1984]
 Book [bookId = 1004,title= Read C++,        price=295.0, author=Henry Harvin,   dop=19/11/1984]
 Book [bookId = 1003,title= Java Programming,price=523.0, author=Gilad Bracha,   dop=23/11/1984]
 Book [bookId = 1002,title= Modern Mainframe,price=295.0, author=Sharad,         dop=19/5/1997]
 Book [bookId = 1001,title= Python Learning, price=715.0, author=Martic C. Brown,dop=2/2/2020]
```

Problem Statement 2: Processing Data with Java SE 8 Streams

Create a Person class with id, name, age, and salary and override all the required methods such as toString, hashcode, equals, and compareTo.

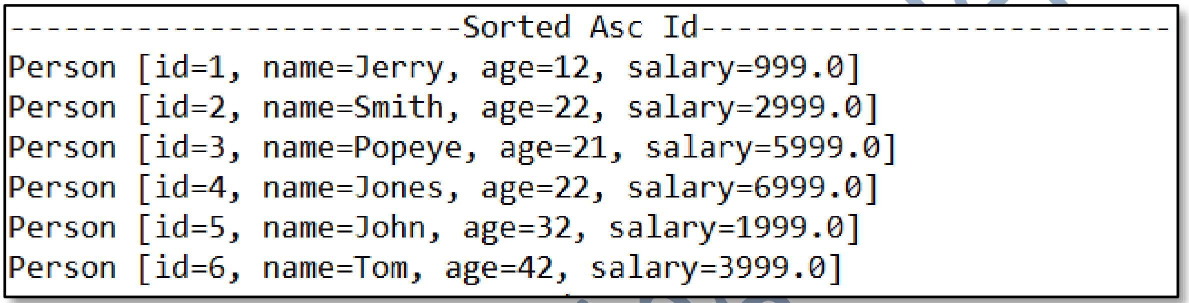
Write a program to accept information of 6 person details and store that in HashSet. Do following operations,

6. Print all the persons details using the Streams and Method Reference features.



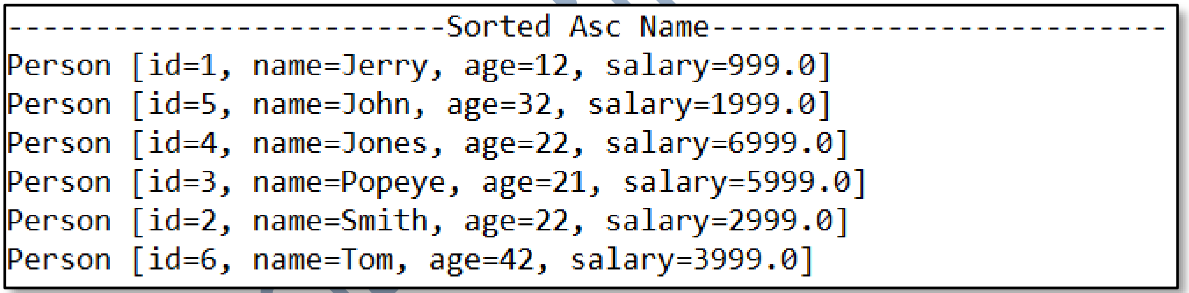
```
------------------------Print all the person records-------------------------
Person [id=4, name=Jones, age=22, salary=6999.0]
Person [id=6, name=Tom, age=42, salary=3999.0]
Person [id=1, name=Jerry, age=12, salary=999.0]
Person [id=5, name=John, age=32, salary=1999.0]
Person [id=2, name=Smith, age=22, salary=2999.0]
Person [id=3, name=Popeye, age=21, salary=5999.0]
```

7. Print all the persons details by sorting the id in ascending order using Comparable and Streams.



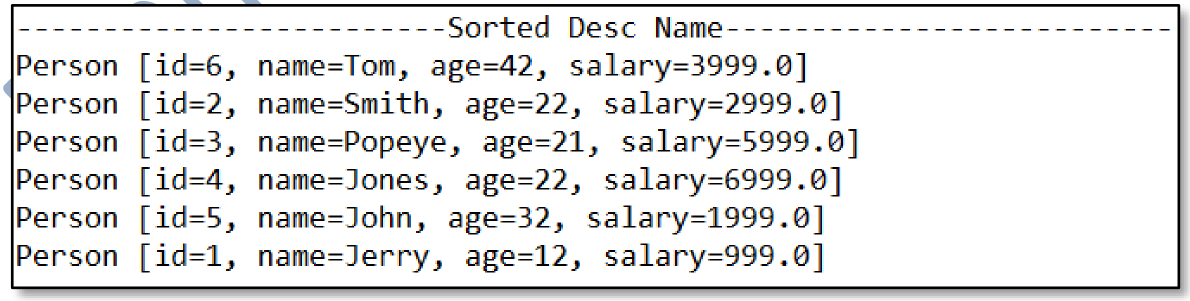
```
-------------------------Sorted Asc Id-------------------------
Person [id=1, name=Jerry, age=12, salary=999.0]
Person [id=2, name=Smith, age=22, salary=2999.0]
Person [id=3, name=Popeye, age=21, salary=5999.0]
Person [id=4, name=Jones, age=22, salary=6999.0]
Person [id=5, name=John, age=32, salary=1999.0]
Person [id=6, name=Tom, age=42, salary=3999.0]
```

8. Print all the persons details by sorting the name in ascending order using Comparator and Streams.



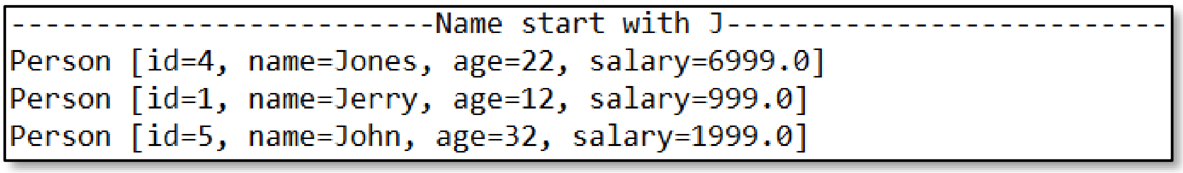
```
-------------------------Sorted Asc Name-------------------------
Person [id=1, name=Jerry, age=12, salary=999.0]
Person [id=5, name=John, age=32, salary=1999.0]
Person [id=4, name=Jones, age=22, salary=6999.0]
Person [id=3, name=Popeye, age=21, salary=5999.0]
Person [id=2, name=Smith, age=22, salary=2999.0]
Person [id=6, name=Tom, age=42, salary=3999.0]
```

9. Print all the persons details by sorting the names in descending order using Comparator and Streams.



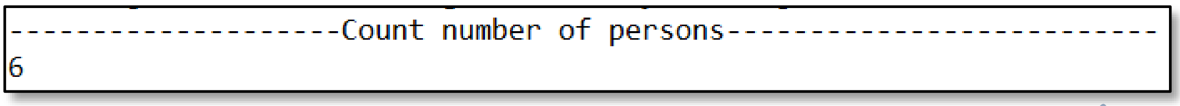
```
-------------------------Sorted Desc Name-------------------------
Person [id=6, name=Tom, age=42, salary=3999.0]
Person [id=2, name=Smith, age=22, salary=2999.0]
Person [id=3, name=Popeye, age=21, salary=5999.0]
Person [id=4, name=Jones, age=22, salary=6999.0]
Person [id=5, name=John, age=32, salary=1999.0]
Person [id=1, name=Jerry, age=12, salary=999.0]
```

10. Print all the persons details whose Name start with J using Streams.



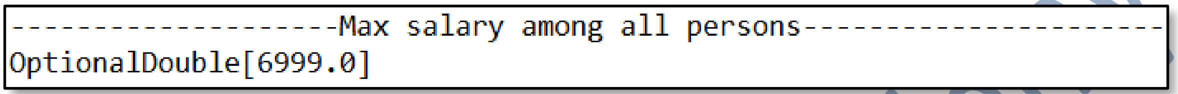
```
-------------------------Name start with J-------------------------
Person [id=4, name=Jones, age=22, salary=6999.0]
Person [id=1, name=Jerry, age=12, salary=999.0]
Person [id=5, name=John, age=32, salary=1999.0]
```

11. Print the count number of persons using Streams.



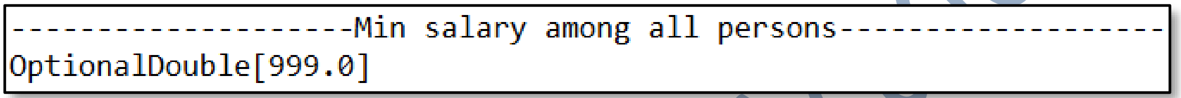
```
--------------------Count number of persons-------------------------
6
```

12. Print the Max salary among all persons using Streams.



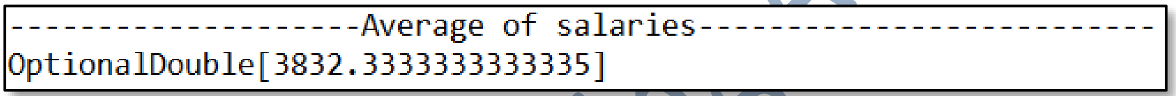
```
--------------------Max salary among all persons---------------------
OptionalDouble[6999.0]
```

13. Print the Min salary among all persons using Streams.



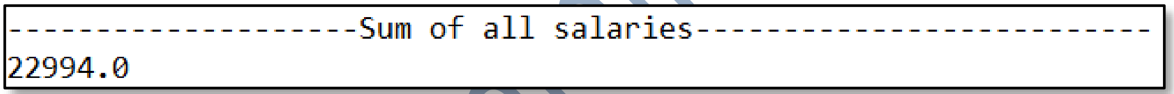
```
--------------------Min salary among all persons-------------------
OptionalDouble[999.0]
```

14. Print the average of all salaries using Streams.



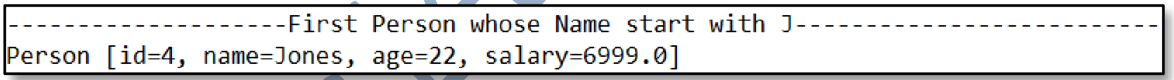
```
--------------------Average of salaries-------------------------
OptionalDouble[3832.3333333333335]
```

15. Print the sum of all salaries using Streams.



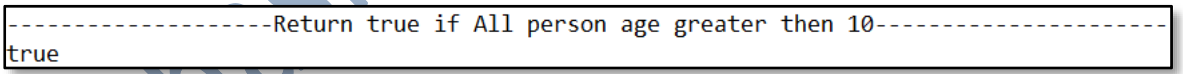
```
--------------------Sum of all salaries-------------------------
22994.0
```

16. Print the First Person whose Name start with J using Streams - filter and findFirst method.



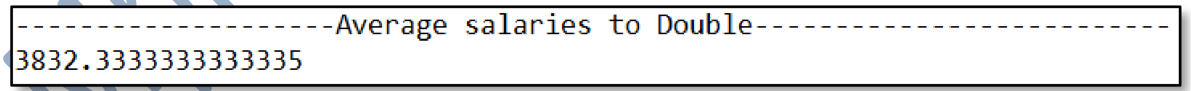
```
-------------------First Person whose Name start with J-------------------------
Person [id=4, name=Jones, age=22, salary=6999.0]
```

17. Check whether all the persons age is greater than 10 using Streams – allMatch method.



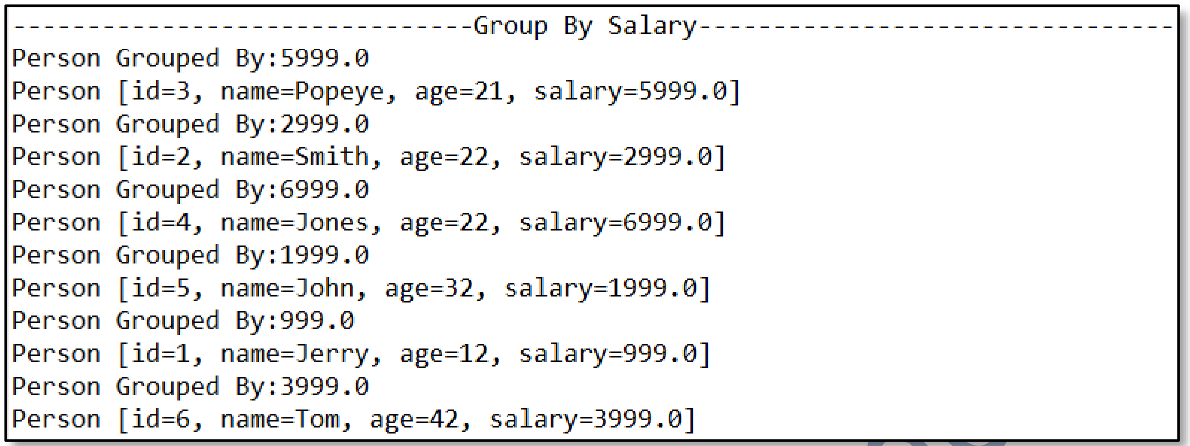
```
-------------------Return true if All person age greater then 10---------------------
true
```

18. Print the average of all salaries using Streams and Collectors.



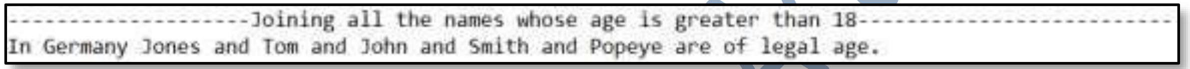
```
--------------------Average salaries to Double-------------------------
3832.3333333333335
```

19. Print all the persons details group by salary using Streams and Collectors.



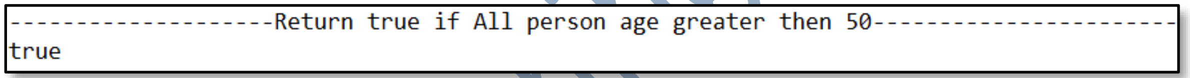
```
-------------------------------Group By Salary---------------------------------
Person Grouped By:5999.0
Person [id=3, name=Popeye, age=21, salary=5999.0]
Person Grouped By:2999.0
Person [id=2, name=Smith, age=22, salary=2999.0]
Person Grouped By:6999.0
Person [id=4, name=Jones, age=22, salary=6999.0]
Person Grouped By:1999.0
Person [id=5, name=John, age=32, salary=1999.0]
Person Grouped By:999.0
Person [id=1, name=Jerry, age=12, salary=999.0]
Person Grouped By:3999.0
Person [id=6, name=Tom, age=42, salary=3999.0]
```

20. Print all the names after joining whose age is greater than 18 using Streams and Collectors.



```
-------------------Joining all the names whose age is greater than 18-------------------------
In Germany Jones and Tom and John and Smith and Popeye are of legal age.
```

21. Check whether all the persons age is greater than 50 using Streams – noneMatch method.



```
--------------------Return true if All person age greater then 50-----------------------
true
```