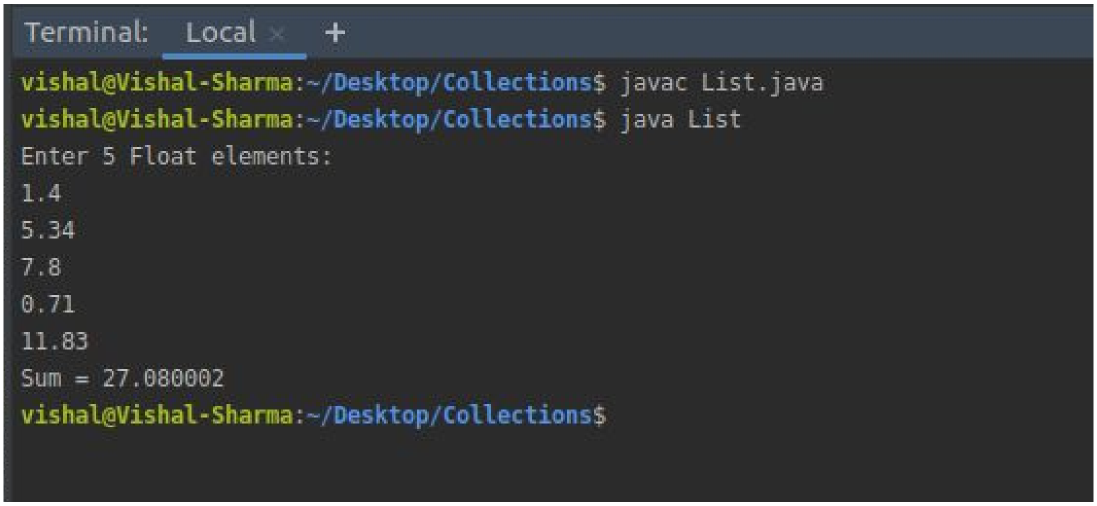
Newer : Vishal Sharma
Newer ID : 4171
Email : vishal.sharma@tothenew.com

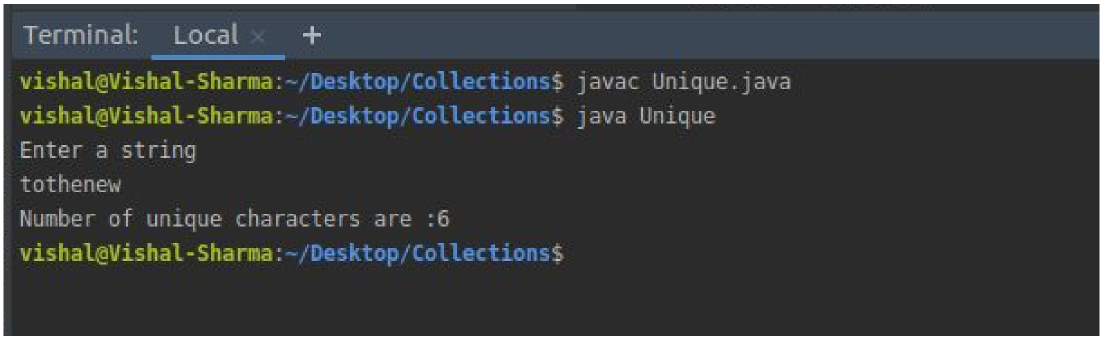
Exercise : Collections in JAVA

1) Write Java code to define List . Insert 5 floating point numbers in List, and using an iterator, find the sum of the numbers in List.
   Answer : Filename - **List.java**



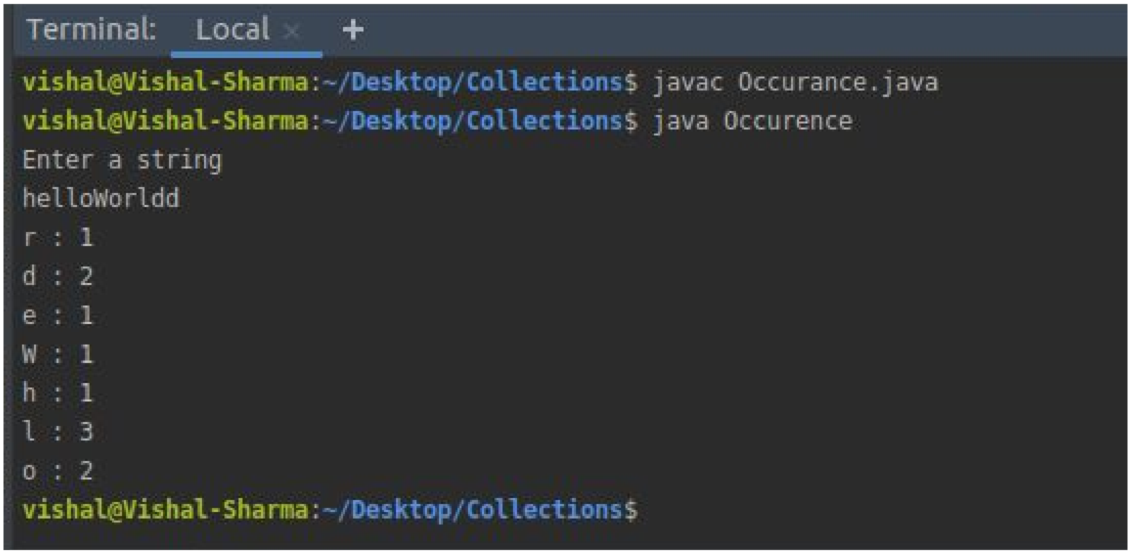
```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac List.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java List
Enter 5 Float elements:
1.4
5.34
7.8
0.71
11.83
Sum = 27.080002
vishal@Vishal-Sharma:~/Desktop/Collections$
```

2) Write a method that takes a string and returns the number of unique characters in the string.
   Answer : Filename - **Unique.java**



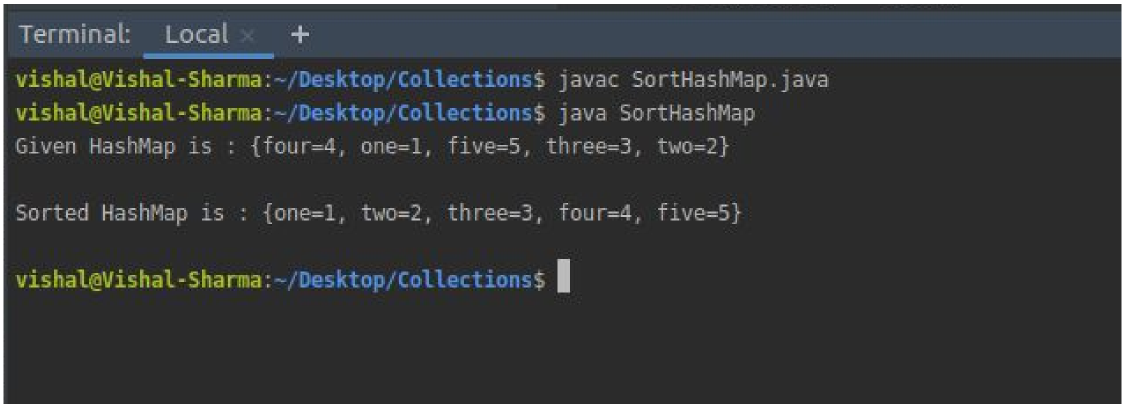
```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac Unique.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java Unique
Enter a string
tothenew
Number of unique characters are :6
vishal@Vishal-Sharma:~/Desktop/Collections$
```

3) Write a method that takes a string and print the number of occurrence of each characters in the string.
Answer : Filename - **<u>Occurance.java</u>**



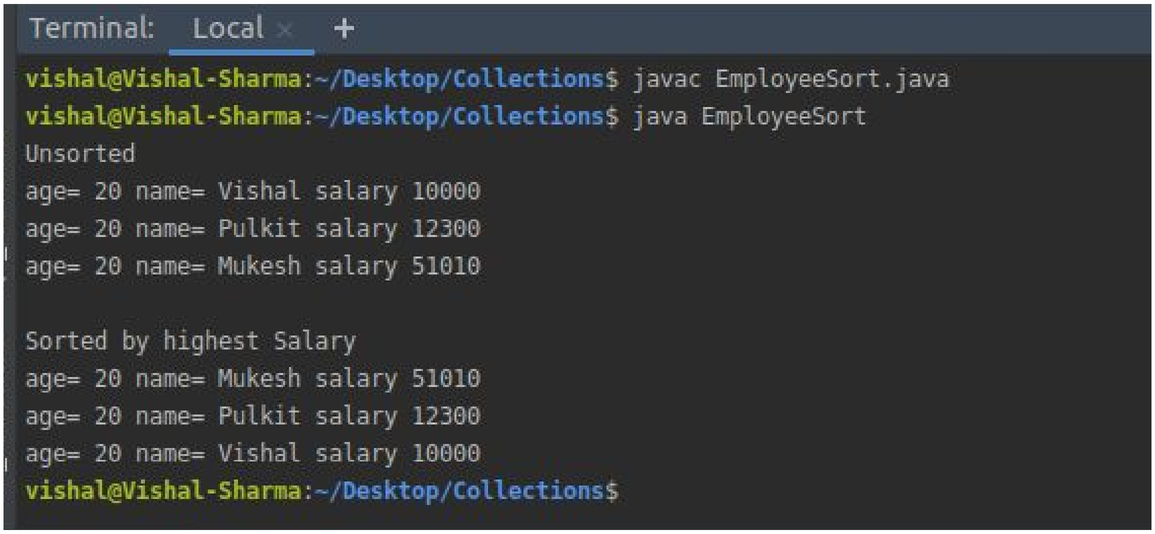
```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac Occurance.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java Occurence
Enter a string
helloWorldd
r : 1
d : 2
e : 1
W : 1
h : 1
l : 3
o : 2
vishal@Vishal-Sharma:~/Desktop/Collections$
```

4) Write a program to sort HashMap by value.
Answer : Filename - **<u>SortHashMap.java</u>**



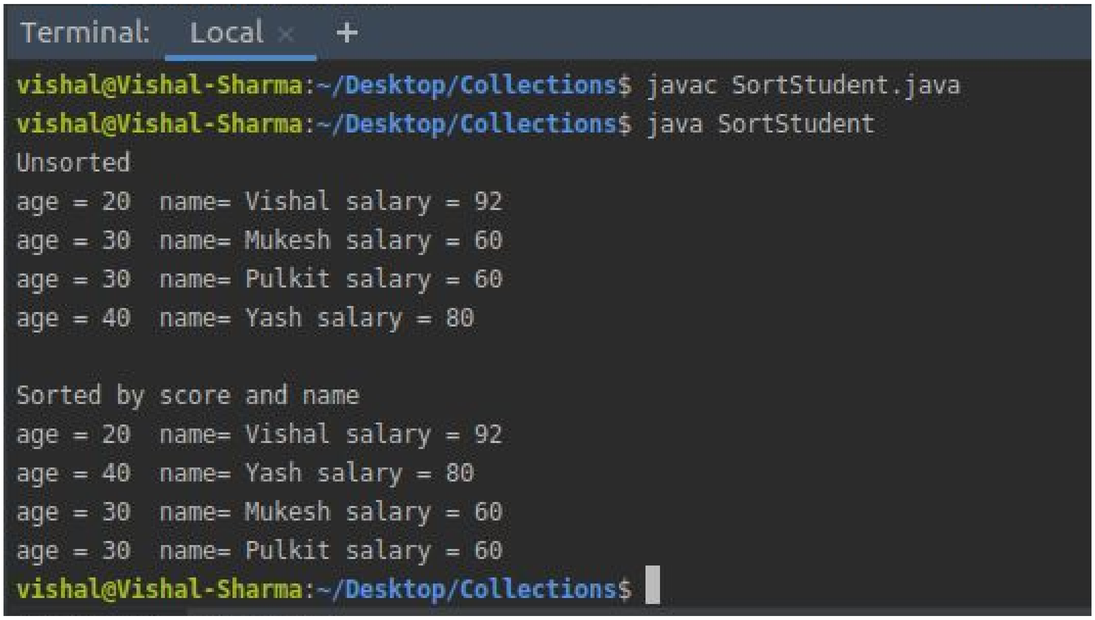
```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac SortHashMap.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java SortHashMap
Given HashMap is : {four=4, one=1, five=5, three=3, two=2}

Sorted HashMap is : {one=1, two=2, three=3, four=4, five=5}

vishal@Vishal-Sharma:~/Desktop/Collections$
```

5) Write a program to sort Employee objects based on highest salary using Comparator. Employee class{ Double Age; Double Salary; String Name}
Answer : Filename - **<u>EmployeeSort.java</u>**



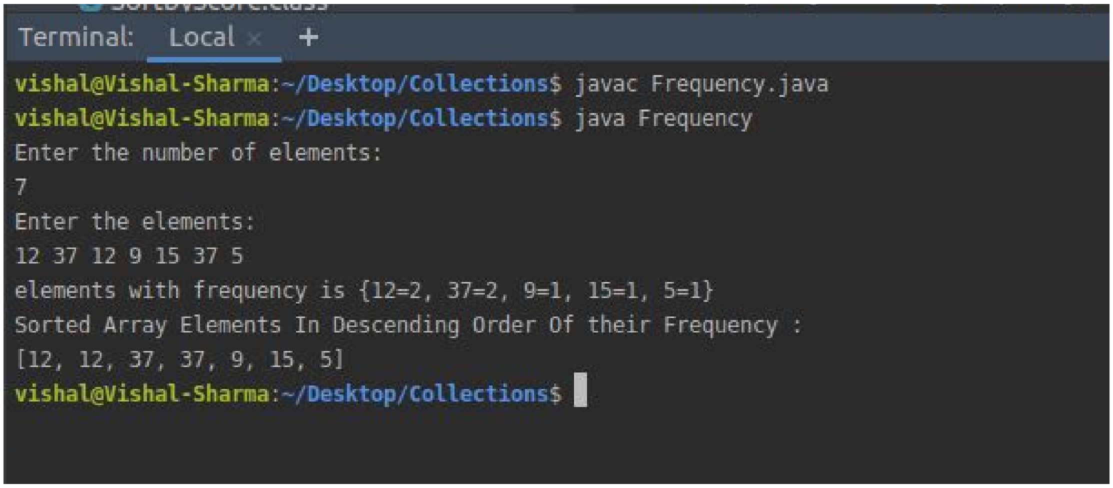
```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac EmployeeSort.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java EmployeeSort
Unsorted
age= 20 name= Vishal salary 10000
age= 20 name= Pulkit salary 12300
age= 20 name= Mukesh salary 51010

Sorted by highest Salary
age= 20 name= Mukesh salary 51010
age= 20 name= Pulkit salary 12300
age= 20 name= Vishal salary 10000
vishal@Vishal-Sharma:~/Desktop/Collections$
```

6) Write a program to sort the Student objects based on Score , if the score are same then sort on First Name . Class Student{ String Name; Double Score; Double Age}
Answer : Filename - **SortStudent.java**



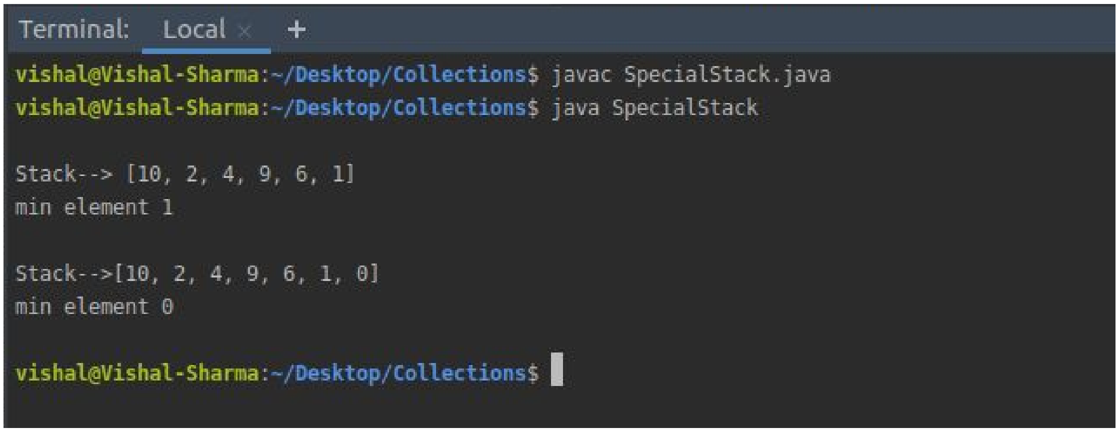
```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac SortStudent.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java SortStudent
Unsorted
age = 20  name= Vishal salary = 92
age = 30  name= Mukesh salary = 60
age = 30  name= Pulkit salary = 60
age = 40  name= Yash salary = 80

Sorted by score and name
age = 20  name= Vishal salary = 92
age = 40  name= Yash salary = 80
age = 30  name= Mukesh salary = 60
age = 30  name= Pulkit salary = 60
vishal@Vishal-Sharma:~/Desktop/Collections$
```

7) Print the elements of an array in the decreasing frequency if 2 numbers have same frequency then print the one which came first.
Answer : Filename - **Frequency.java**



```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac Frequency.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java Frequency
Enter the number of elements:
7
Enter the elements:
12 37 12 9 15 37 5
elements with frequency is {12=2, 37=2, 9=1, 15=1, 5=1}
Sorted Array Elements In Descending Order Of their Frequency :
[12, 12, 37, 37, 9, 15, 5]
vishal@Vishal-Sharma:~/Desktop/Collections$ 
```

8) Design a Data Structure SpecialStack that supports all the stack operations like push(), pop(), isEmpty(), isFull() and an additional operation getMin() which should return minimum element from the SpecialStack. (Expected complexity O(1))
Answer : Filename - **StackOperations.java**



```
vishal@Vishal-Sharma:~/Desktop/Collections$ javac SpecialStack.java
vishal@Vishal-Sharma:~/Desktop/Collections$ java SpecialStack

Stack--> [10, 2, 4, 9, 6, 1]
min element 1

Stack-->[10, 2, 4, 9, 6, 1, 0]
min element 0

vishal@Vishal-Sharma:~/Desktop/Collections$ 
```