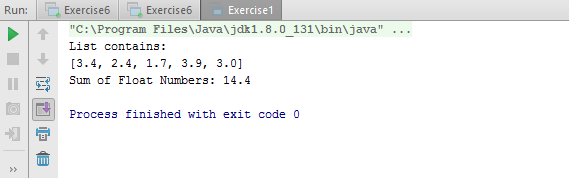
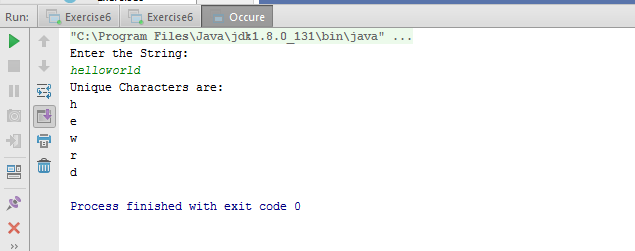
1. **Write Java code to define List . Insert 5 floating point numbers in List, and using an iterator, find the sum of the numbers in List.**

**import** java.util.\*;  
**public class** Exercise1 {  
  
 **public static void** main(String[] args) {  
 **float** sum = 0;  
 List<Float> list = **new** ArrayList<Float>();  
 list.add(**new** Float(3.4));  
 list.add(**new** Float(2.4));  
 list.add(**new** Float(1.7));  
 list.add(**new** Float(3.9));  
 list.add(**new** Float(3.0));  
  
 Iterator<Float> itr = list.iterator();  
 System.***out***.println(**"List contains: "**);  
 System.***out***.println(list);  
 **while**(itr.hasNext())  
 {  
 Float num = itr.next();  
 sum += num;  
 }  
  
 System.***out***.println(**"Sum of Float Numbers: "**+sum);  
  
 }  
}



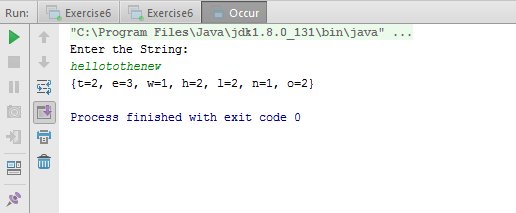
1. **Write a method that takes a string and returns the number of unique characters in the string.**

**import** java.util.\*;  
  
**public class** Exercise2 {  
  
 **void** occurence (String str) {  
 Map<Character, Integer> occ = **new** LinkedHashMap<Character, Integer>();  
 **for** (**char** c : str.toCharArray()) {  
  
 **if** (!occ.containsKey(c)) {  
 occ.put(c, 1);  
 } **else** {  
 occ.put(c, occ.get(c) + 1);  
 }  
  
 }  
 System.***out***.println(**"Unique Characters are: "**);  
 **for** (Map.Entry<Character,Integer> entry : occ.entrySet()) {  
  
 **if** (entry.getValue() == 1)  
 System.***out***.println(entry.getKey());  
 }  
 }  
}  
  
**class** Occure  
{  
 **public static void** main(String[] args) {  
 Exercise2 obj = **new** Exercise2();  
 System.***out***.println(**"Enter the String: "**);  
 Scanner sc = **new** Scanner(System.***in***);  
 String str = sc.nextLine();  
 obj.occurence(str);  
 }  
}



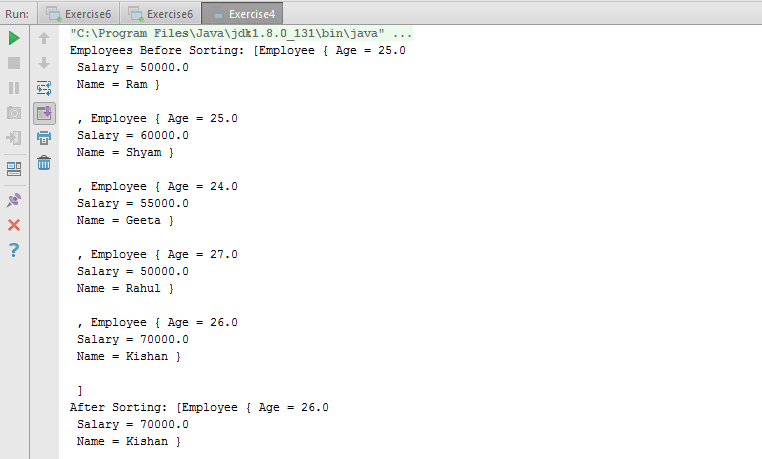
1. **Write a method that takes a string and print the number of occurrence of each character characters in the string.**

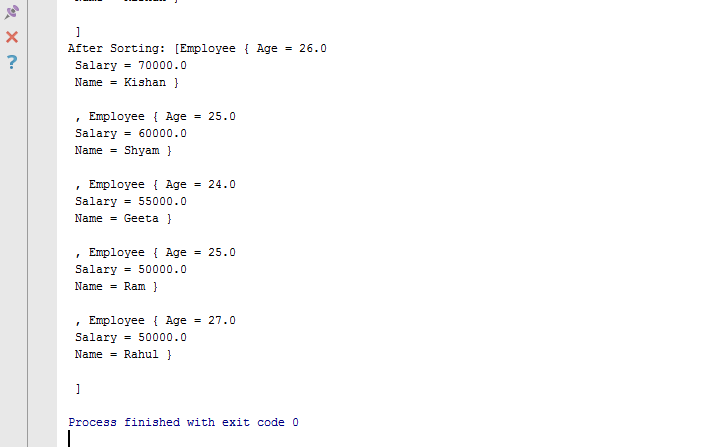
**import** java.util.\*;  
  
**public class** Exercise3 {  
  
 **void** occurence (String str)  
 {  
 Map<Character,Integer> occ = **new** HashMap<Character,Integer>();  
 **for**(**char** c : str.toCharArray())  
 {  
  
 **if**(!occ.containsKey(c))  
 {  
 occ.put(c,1);  
 }  
 **else** {  
 occ.put(c,occ.get(c)+1);  
 }  
  
 }  
 System.***out***.println(occ);  
 }  
  
  
}  
  
  
**class** Occur  
{  
 **public static void** main(String[] args) {  
 Exercise3 obj = **new** Exercise3();  
 System.***out***.println(**"Enter the String: "**);  
 Scanner sc = **new** Scanner(System.***in***);  
 String str = sc.nextLine();  
 obj.occurence(str);  
 }  
}



1. **Write a program to sort Employee objects based on highest salary using Comparator. Employee class{ Double Age; Double Salary; String Name**

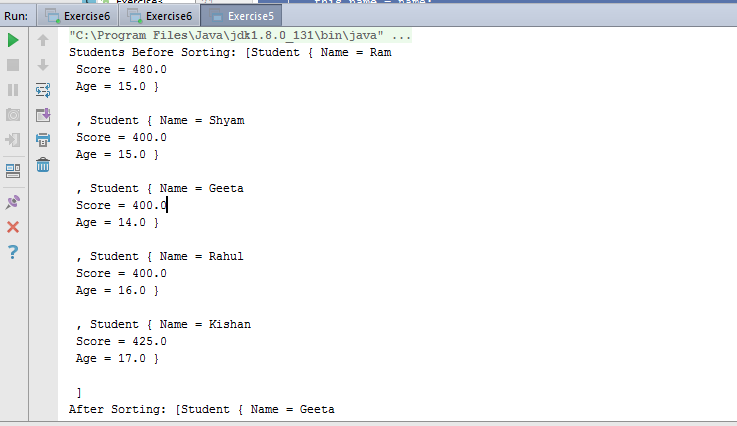
**import** java.util.\*;  
  
**class** Employee {  
 **double age**;  
 **double salary**;  
 String **name**;  
  
 **public double** getAge() {  
 **return age**;  
 }  
  
 **public double** getSalary() {  
 **return salary**;  
 }  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public** Employee(**double** age, **double** salary, String name) {  
 **this**.**age** = age;  
 **this**.**salary** = salary;  
 **this**.**name** = name;  
 }  
  
 **public static final** Comparator<Employee> ***SalaryComparator*** = **new** Comparator<Employee>() {  
 @Override  
 **public int** compare(Employee o1, Employee o2) {  
 **return** (**int**) (o2.getSalary() - o1.getSalary());  
 }  
  
  
 };  
  
 **public** String toString()  
 {  
 **return "Employee { Age = "**+**age**+**"\n Salary = "**+**salary** +**"\n Name = "**+**name**+ **" }\n\n "**;  
 }  
  
  
}  
**public class** Exercise4 {  
 **public static void** main(String[] args) {  
 Employee[] obj = **new** Employee[5];  
 obj[0] = **new** Employee(25,50000,**"Ram"**);  
 obj[1] = **new** Employee(25,60000,**"Shyam"**);  
 obj[2] = **new** Employee(24,55000,**"Geeta"**);  
 obj[3] = **new** Employee(27,50000,**"Rahul"**);  
 obj[4] = **new** Employee(26,70000,**"Kishan"**);  
  
 List <Employee> list = **new** ArrayList<Employee>();  
 list.add(obj[0]);  
 list.add(obj[1]);  
 list.add(obj[2]);  
 list.add(obj[3]);  
 list.add(obj[4]);  
 System.***out***.println(**"Employees Before Sorting: "**+list);  
 Collections.*sort*(list,Employee.***SalaryComparator***);  
 System.***out***.println(**"After Sorting: "**+list);  
 }  
  
}

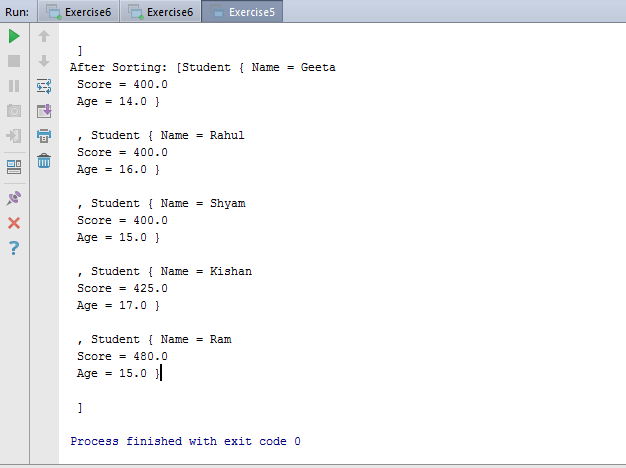




1. **Write a program to sort the Student objects based on Score , if the score are same then sort on First Name . Class Student{ String Name; Double Score; Double Age**

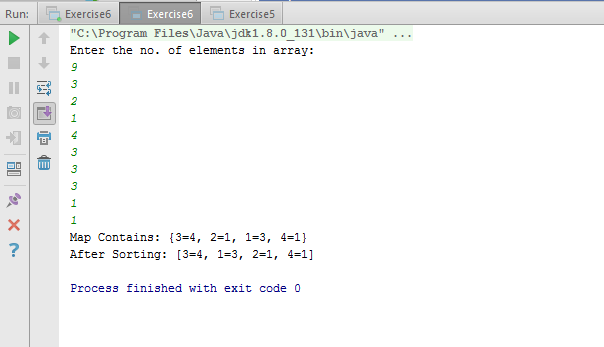
**import** java.util.\*;  
**class** Student {  
 String **name**;  
 **double score**;  
 **double age**;  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public double** getScore() {  
 **return score**;  
 }  
  
 **public double** getAge() {  
 **return age**;  
 }  
  
 Student (String name, **double** score, **double** age)  
 {  
 **this**.**name** = name;  
 **this**.**score** = score;  
 **this**.**age** = age;  
  
 }  
  
 **public** String toString()  
 {  
 **return "Student { Name = "**+**name**+**"\n Score = "**+**score** +**"\n Age = "**+**age**+ **" }\n\n "**;  
 }  
  
  
  
 **public static final** Comparator<Student> ***ScoreandNameComparator*** = **new** Comparator<Student>() {  
 @Override  
 **public int** compare(Student o1, Student o2) {  
 **int** flag = (**int**)(o1.getScore()- o2.getScore());  
  
 **if**(flag == 0)  
 {  
 flag = o1.getName().compareTo( o2.getName());  
 }  
  
 **return** flag;  
 }  
 };  
}  
  
**public class** Exercise5 {  
 **public static void** main(String[] args) {  
 Student[] obj = **new** Student[5];  
 obj[0] = **new** Student(**"Ram"**,480,15);  
 obj[1] = **new** Student(**"Shyam"**,400,15);  
 obj[2] = **new** Student(**"Geeta"**,400,14);  
 obj[3] = **new** Student(**"Rahul"**,400,16);  
 obj[4] = **new** Student(**"Kishan"**,425,17);  
  
 List <Student> list = **new** ArrayList<Student>();  
 list.add(obj[0]);  
 list.add(obj[1]);  
 list.add(obj[2]);  
 list.add(obj[3]);  
 list.add(obj[4]);  
 System.***out***.println(**"Students Before Sorting: "**+list);  
 Collections.*sort*(list,Student.***ScoreandNameComparator***);  
 System.***out***.println(**"After Sorting: "**+list);  
 }  
  
}





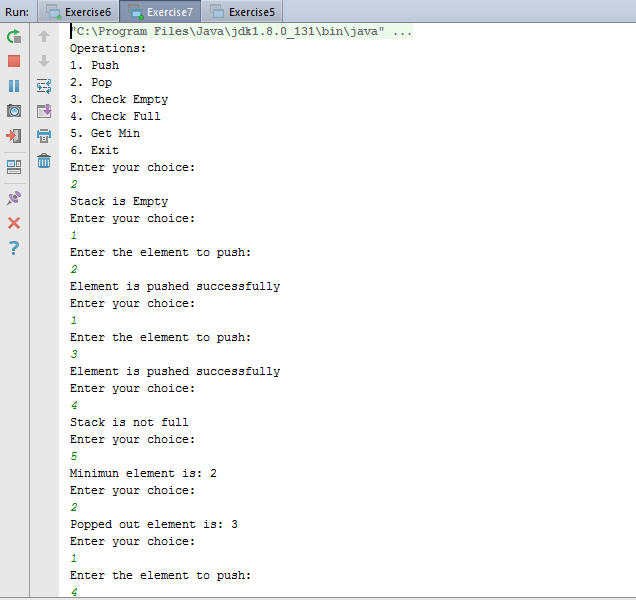
1. **Print the elements of an array in the decreasing frequency if 2 numbers have same frequency then print the one which came first.**

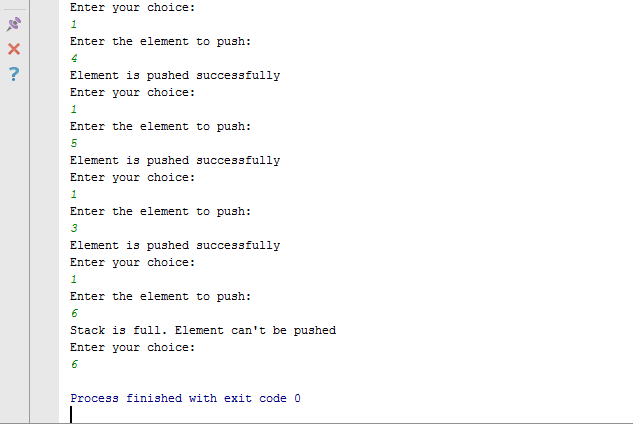
**import** java.util.\*;  
  
**class** Freq {  
  
 **public static final** Comparator<Map.Entry<Integer,Integer>> ***FrequencyComparator*** = **new** Comparator<Map.Entry<Integer,Integer>>() {  
 @Override  
 **public int** compare(Map.Entry<Integer,Integer> o1, Map.Entry<Integer,Integer> o2) {  
 **return** (o2.getValue() - o1.getValue());  
 }  
 };  
}  
**public class** Exercise6 {  
 **public static void** main(String[] args) {  
 Freq obj = **new** Freq();  
 System.***out***.println(**"Enter the no. of elements in array: "**);  
 Scanner sc = **new** Scanner(System.***in***);  
 Integer size = sc.nextInt();  
 **int** ele;  
 Map<Integer, Integer> occ = **new** LinkedHashMap<Integer,Integer>();  
 **for** (**int** i = 0; i < size; i++) {  
 ele = sc.nextInt();  
  
 **if** (!occ.containsKey(ele)) {  
 occ.put(ele, 1);  
 } **else** {  
 occ.put(ele, occ.get(ele) + 1);  
 }  
 }  
  
 System.***out***.println(**"Map Contains: "**+occ);  
 List list = **new** ArrayList(occ.entrySet());  
 Collections.*sort*(list,Freq.***FrequencyComparator***);  
 System.***out***.println(**"After Sorting: "**+list);  
  
 }  
}



1. **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))**

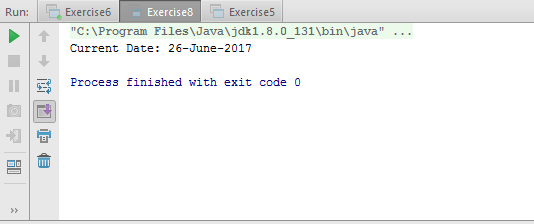
**import** java.util.\*;  
  
**import static** java.lang.System.*exit*;  
  
**class** SpecialStack {  
  
 **private int stackSize**;  
  
 Stack<Integer> **stackobj** = **new** Stack<Integer>();  
 Stack<Integer> **minStack** = **new** Stack<Integer>();  
  
 SpecialStack(**int** size){  
 **this**.**stackSize** = size;  
 }  
  
 **void** push(**int** ele) {  
 **if** (!isFull()) {  
 **stackobj**.push(ele);  
 **if** (**minStack**.empty()) {  
 **minStack**.push(ele);  
  
 } **else** {  
 **if** (**minStack**.peek() >= ele) {  
 **minStack**.push(ele);  
 } **else** {  
 **minStack**.push(**minStack**.peek());  
 }  
 }  
  
 System.***out***.println(**"Element is pushed successfully"**);  
 **stackSize**--;  
 }  
 **else**{  
 System.***out***.println(**"Stack is full. Element can't be pushed"**);  
 }  
  
 }  
  
 **void** pop() {  
 **if** (!**stackobj**.empty()) {  
 **minStack**.pop();  
 **int** ele = **stackobj**.pop();  
 System.***out***.println(**"Popped out element is: "**+ele);  
 }  
 **else**{  
 System.***out***.println(**"Stack is Empty"**);  
 }  
 }  
  
 **boolean** isFull(){  
 **if**(**stackSize** > 0){  
 **return false**;  
 }  
 **else**{  
 **return true**;  
 }  
 }  
  
 **void** isEmpty() {  
 **if** (**stackobj**.empty()) {  
 System.***out***.println(**"Stack is Empty"**);  
  
 }  
 **else**{  
 System.***out***.println(**"Stack contains element"**);  
 }  
 }  
  
  
 **void** getMin() {  
  
 **if** (!**stackobj**.empty()) {  
 **int** ele = **minStack**.peek();  
 System.***out***.println(**"Minimun element is: "**+ele);  
 }  
  
 **else**{  
 System.***out***.println(**"Stack is empty"**);  
 }  
 }  
}  
  
**public class** Exercise7{  
 **public static void** main(String[] args) {  
  
 **int** c;  
 SpecialStack obj = **new** SpecialStack(5);  
  
 System.***out***.println(**"Operations: "**);  
 System.***out***.println(**"1. Push"**);  
 System.***out***.println(**"2. Pop"**);  
 System.***out***.println(**"3. Check Empty"**);  
 System.***out***.println(**"4. Check Full"**);  
 System.***out***.println(**"5. Get Min"**);  
 System.***out***.println(**"6. Exit"**);  
 **do**{  
 System.***out***.println(**"Enter your choice: "**);  
 Scanner sc = **new** Scanner(System.***in***);  
 c = sc.nextInt();  
  
 **switch** (c) {  
 **case** 1:  
 System.***out***.println(**"Enter the element to push: "**);  
 **int** ele = sc.nextInt();  
 obj.push(ele);  
 **break**;  
  
 **case** 2:  
 obj.pop();  
 **break**;  
  
 **case** 3:  
 obj.isEmpty();  
 **break**;  
  
 **case** 4:  
 **boolean** full = obj.isFull();  
 **if**(full == **true**)  
 System.***out***.println(**"Stack is Full"**);  
 **else** System.***out***.println(**"Stack is not full"**);  
 **break**;  
  
 **case** 5:  
 obj.getMin();  
 **break**;  
  
 **case** 6:  
 *exit*(0);  
  
 **default**:  
 System.***out***.println(**"Entered choice is wrong"**);  
 }  
 }**while**(c != 6 );  
 }  
}





1. **Write a program to format date as example "21-March-2016"**

**import** java.text.DateFormat;  
**import** java.text.SimpleDateFormat;  
**import** java.util.\*;  
**public class** Exercise8 {  
 **public static void** main(String[] args) {  
 DateFormat dateFormat = **new** SimpleDateFormat(**"dd-MMMM-yyyy"**);  
 Date date = **new** Date();  
 System.***out***.println(**"Current Date: "**+dateFormat.format(date));  
  
 }  
}



1. **Write a program to display times in different country format.**

**import** java.text.DateFormat;  
**import** java.text.SimpleDateFormat;  
**import** java.util.\*;  
  
**import static** java.util.Locale.\*;  
  
**public class** Exercise9 {  
 **public static void** main(String[] args) {  
 DateFormat dateFormat = **new** SimpleDateFormat(**"dd-MMM-yyyy HH:mm:ss"**);  
 Date date = **new** Date();  
 System.***out***.println(**"Current Date and Time: "**+dateFormat.format(date));  
  
 String [] zones = {**"Africa/Dakar"**,**"Asia/Kolkata"**,**"America/Los\_Angeles"**,**"Europe/Copenhagen"**};  
 **for**(String zone:zones ) {  
 dateFormat.setTimeZone(TimeZone.*getTimeZone*(zone));  
 String time = dateFormat.format(date);  
 System.***out***.println(**"Time at "**+zone+**" is : "**+time);  
 }  
 System.***out***.println(**"\n"**);  
 Locale [] locales = {***US***, ***UK***, ***GERMANY***, ***FRANCE***,***CANADA***,***CHINA***,***ITALY***,***JAPAN***};  
 **for**(Locale locale: locales){  
 System.***out***.println(**"The Date for "**+locale.getDisplayCountry()+**": "**);  
 dateFormat = DateFormat.*getDateInstance*(DateFormat.***FULL***, locale);  
 System.***out***.println(dateFormat.format(date));  
 }  
 }  
}

