1.

파이썬:

n = int(input()) # 회원수  
userList = [] # 회원 리스트  
for i in range(n):  
 age, name = input().split() # 공백을 기준으로 나이와 이름을 받아온다  
 userList.append([int(age),name, i]) # 0번째 인덱스: 나이, 1번째 인덱스: 이름으로 저장한다, 2: 회원가입 순서  
userList.sort(key =lambda x:(x[0], x[2])) # 0번째인 나이를 기준으로 오름차순, 같다면 회원가입순서로 정렬  
for age, name, id in userList:  
 print('{} {}'.format(age, name)) # 형식에 맞게 출력

자바:

import java.util.Arrays;  
import java.util.Comparator;  
import java.util.List;  
import java.util.Scanner;  
  
// 이름과 나이를 갖는 유저 객체를 생성하기  
class User{  
 private String name; // 이름  
 private int age; // 나이  
 private int idNum; // 가입 순서  
  
 // 생성자  
 public User(int age, String name, int idNum) {  
 this.name = name;  
 this.age = age;  
 this.idNum = idNum;  
 }  
 // private 이기 떄문에 출력 메소드 따로 지정해준다  
 public void UserPrint() {  
 System.*out*.println(age+""+name);  
 }  
  
 public int getIdNum() {  
 return idNum;  
 }  
  
 public int getAge() {  
 return age;  
 }  
}  
  
public class OrderByAge {  
 public static void main(String[] args) {  
 String name;  
 int age;  
 Scanner sc = new Scanner(System.*in*);  
 int n = sc.nextInt(); // 회원의 수  
 User [] users = new User[n];  
 for (int i = 0; i < n; i++) {  
 age = sc.nextInt();  
 name = sc.nextLine();  
 users[i] = new User(age,name,i); // id 는 회원 가입 순서대로 객체를 생성한다  
 }  
 List<User> usersSort = Arrays.*asList*(users);  
 System.*out*.println(usersSort);  
 // 리스트 타입으로 객체를 리스트로 변한한뒤 Comparator을 이용하여 나이순대로 정렬후 다음 가입 순서 대로 정렬한다  
 usersSort.sort(Comparator.*comparing*(User::getAge).thenComparing(User::getIdNum));  
 for (User user : users) {  
 user.UserPrint();  
 }  
 }  
}

리뷰 : 평소 문제풀이식 코딩은 파이썬으로만 접하고 있었는데 이번을 기회로 자바를 연습하고 싶어 자바를 통하여 구현을 해보았습니다

평소와는 다른 방식으로 객체를 생성하고 각 유저들을 객체로 저장하며 캡슐화 한다는것의 이점을 살리는것에 목적을 두고 문제에 접근하였습니다

자바 공부도 앞으로 게을리 하지않고 열심히 하여 서브 언어가 아닌 메인 언어급으로 성장시켜야할 필요성을 느꼈습니다.

2.

파이썬:

from collections import Counter  
n = int(input()) # 수들의 수  
numbers = [int(input()) for \_ in range(n)] # n개의 수를 리스트에 저장  
# 산술평균  
print(round(sum(numbers)/n)) # 소수점 1의 자리에서 반올림한 평균값  
# 중앙값  
numbers.sort() # 수들을 오름차순으로 정렬  
print(numbers[n//2]) # 오름차순으로 정렬된 리스트의 중앙값  
# 최빈값  
countNum = Counter(numbers) # 키:수, 벨류:빈도수 형식의 딕셔너리 생성  
maxFreqNum = max(countNum.values()) # 빈도수들중 가장 큰값  
maxFreqCnt = list(countNum.values()).count(maxFreqNum) # 최대빈도수를 가진 값이 많은지 체크  
if maxFreqCnt >1: # 최대빈도수를 가진 값이 1보다 크다면  
 isSecond = False  
 for key in sorted(countNum.keys()): # 오름차순으로 정렬된 키값들  
 if isSecond: # 만약 두번째로 작은 최대빈도수 값이라면 그 수를 출력하고 반복 종료  
 print(key)  
 break  
 if countNum[key] == maxFreqNum: # 해당 키의 벨류값이 최대 빈도수라면 isSecond를 True로 바꾼다  
 isSecond = True  
else: # 최대빈도수를 가진 값이 하나라면 그 값만 찾고 반복 종료  
 for key in countNum.keys():  
 if countNum[key]==maxFreqNum:  
 print(key)  
 break  
# 범위  
print(numbers[-1]-numbers[0]) # 최대값과 최솟값의 차이

자바:

import java.util.\*;  
  
public class statisticsSort {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 int n = sc.nextInt(); // 수의 개수  
 int[] numbers = new int[n];  
 for (int i = 0; i < n; i++) {  
 numbers[i] = sc.nextInt(); // 배열에 숫자 입력 받고 담아준다  
 }  
 double avg = Arrays.*stream*(numbers).sum()/(n\*1.0);  
 // 산술 평균  
 System.*out*.printf("%.0f\n",avg); // 소수점 1의 자리에서 반올림  
 // 중앙 값  
 Arrays.*sort*(numbers);  
 System.*out*.println(numbers[n/2]); // n은 항상 홀수이기에 중앙값은 n/2 가 됨  
 // 최빈 값  
 HashMap<Integer,Integer> countNum = new HashMap<>(); // 키: 수 , 벨류 : 빈도수  
 for (int number : numbers) {  
 if (countNum.get(number)==null){  
 countNum.put(number, 1); // 만약 해당 수를 카운트 하는게 처음이라면  
 }  
 else {  
 countNum.put(number,countNum.get(number)+1); // 해당 수가 존재한다면  
 }  
 }  
 Integer maxVal = Collections.*max*(countNum.values()); // 빈도수중 max 값을 찾는다  
 int maxFreq = Collections.*frequency*(countNum.values(),maxVal); // 빈도수 max값이 여러개 존재하는지 확인  
 // maxFreq 가 1을 초과할 경우 최빈값중 두번째로 작은 값을 출력하게 한다  
 Object[] sortedKey = countNum.keySet().toArray();  
 Arrays.*sort*(sortedKey); // 키값을 기준으로 정렬해준다  
 if (maxFreq > 1){  
 boolean isSecond = false;  
 for (Object target : sortedKey) {  
 if(isSecond){  
 // 만약 최대빈도수 이면서 두번째로 작은 값이라면 isSecond는 True 가 된다  
 System.*out*.println(target);  
 break; // 그뒤는 고려할 필요가 없으므로 탈출  
 }  
 if(countNum.get(target)==maxVal){  
 isSecond = true;  
 continue;  
 // 만약 타겟값의 빈도수가 최대 빈도수와 같다면 isSecond를 True로 바꿔준다  
 }  
 }  
 }else {  
 for (Integer target : countNum.keySet()) {  
 if(countNum.get(target)==maxVal){  
 System.*out*.println(target);  
 break; // 하나 밖에 존재하지 않는다면 최대빈도수를 가진 수만 출력하고 종료  
 }  
 }  
 }  
 // 범위  
 // 이미 정렬되어진 수들이기 때문에 가장 뒤에있는 수 - 가장 앞에있는 수를 하면 범위를 알 수 있다  
 int bound = numbers[numbers.length-1]-numbers[0];  
 System.*out*.println(bound);  
 }  
}

리뷰 : 자바에는 Counter 라이브러리가 있는지 모르겠지만 변수타입이 다양하다 보니 헤메는 시간이 많았던것같습니다. 같은 부류의 정렬이지만

하는 법이 다르기 때문에 학습에도 좋은 기회가 되었던것 같습니다.