1. Strings – Unique & Existing Characters

Obtain two strings from user as input. Your program should modify the first string such that all the characters are replaced by plus sign (+), except the characters that are present in the second string.

That is, if one or more characters of first string appear in second string, they will not be replaced by plus sign (+).

Return the modified string as output. Note - Ignore case.

Include a class UserMainCode with a static method **replacePlus,** which accepts two string variables. The return type is the modified string.

Create a Class Main that would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings with a maximum size of 100 characters.

Output consists of a single string.

Refer the sample output for formatting specifications.

Sample Input 1:

abcxyz

axdef

Sample Output 1:

a++ x++

Sample Input 2:

ABCDEF

feCBAd

Sample Output 2:

ABCDEF

2. Unique Even Sum

Write a program to read an array, eliminiate duplicate elements, and calculate the sum of even numbers (values) present in the array.

Include a class UserMainCode with a static method **addUniqueEven** that accepts a single integer array. The return type (integer) should be the sum of the even numbers. In case there is no even number, it should return -1.

Create a Class Main that would be used to accept input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next "n" integers correspond to the elements in the array.

In case there is no even integer in the input array, print **no even numbers** as output. Otherwise, print the sum.

Refer the sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20.

Sample Input 1:

4

2

5

1

4

Sample Output 1:

6

Sample Input 2:

3

1

1

1

Sample Output 2:

no even numbers

3. String Occurences

Obtain two strings from user as input. Your program should count the number of occurences of second word of second sentence in the first sentence.

Return the count as output. Note - Consider case.

Include a class UserMainCode with a static method **countNoOfWords** that accepts two string variables. The return type is the modified string.

Create a Class Main that would be used to accept two input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings with a maximum size of 100 characters.

Output consists of a single string.

Refer the sample output for formatting specifications.

Sample Input 1:

abc bcd abc bcd abc abc

av abc

Sample Output 1:

4

Sample Input 2:

ABC xyz AAA

w abc

Sample Output 2:

0

4. ArrayList Manipulation

Write a program that performs the following actions:

1. Read 2n integers as input.
2. Create two array lists to store n elements in each array list.
3. Write a function **generateOddEvenList** that accepts these two array lists as input.
4. The function fetches the odd index elements from first array list and even index elements from second array list and adds them to a new array list according to their index.
5. Return the arraylist.

Include a class UserMainCode with the static method **generateOddEvenList** that accepts two arraylists and returns an arraylist.

Create a Class Main that would be used to read 2n integers and call the static method present in UserMainCode.

Note:

- The index of first element is 0.

- Consider 0 an even number.

- Maintain order in the output array list

Input and Output Format:

Input consists of 2n+1 integers. The first integer denotes the size of the arraylist, the next n integers are values of the first arraylist, and the last n integers are values of the second arraylist.

Output consists of a modified arraylist as per step 4.

Refer sample output for formatting specifications.

Sample Input 1:

5

12

13

14

15

16

2

3

4

5

6

Sample Output 1:

2

13

4

15

5. Mastering Hashmap

You have recently learnt about HashMaps, and in order to master them, you try and use them in all of your programs.

Your trainer/teacher has given you the following exercise:

1. Read 2n numbers as input, where the first number represents a key and second number represents a value. Both the numbers are of integer type.
2. Write a function **getAverageOfOdd** to find out average of all values whose keys are represented by odd numbers. Assume the average is an integer and never a decimal number. Return the average as output. Include this function in class UserMainCode.

Create a Class Main that would be used to read 2n numbers and build the hashmap. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+ 1 integers. The first integer specifies the value of n (essentially the hashmap size). The next pair of n numbers denote the key and value.

Output consists of an integer representing the average.

Refer sample output for formatting specifications.

Sample Input 1:

4

2

34

1

4

5

12

4

22

Sample Output 1:

8

6. Anagram

Write a program to check whether the two given strings are anagrams.

Note: Rearranging the letters of a word or phrase to produce a new word or phrase using all the original letters exactly once is called “Anagram."

Include a class **UserMainCode** with a static method **“getAnagram”** that accepts 2 strings as arguments and returns an integer. The method returns 1 if the 2 strings are anagrams. Else, it returns -1.

Create a Class **Main** that would get 2 Strings as input and call the static method **getAnagram** present in the UserMainCode.

Input and Output Format:
Input consists of 2 strings. Assume that all characters in the string are lower case letters.
Output consists of a string that is either “Anagram” or “Not Anagram.”

Sample Input 1:
eleven plus two
twelve plus one

Sample Output 1:
Anagrams

Sample Input 2:
orchestra
carthorse

Sample Output 2:
Anagrams

Sample Input 3:
hello
world

Sample Output 3:
Not Anagrams

7. Retirement

Consider an input, HashMap, that contains key as the ID and DOB as value of employees. Write a program to find out employees eligible for retirement. A person is eligible for retirement if his or her age is greater than or equal to 60.

Assume that the current date is 01/01/2014.

Include a class **UserMainCode** with a static method “retirementEmployeeList” that accepts a HashMap<String,String> as input and returns an ArrayList<String>. In this method, add the Employee IDs of all the persons eligible for retirement to list and return the sorted list. (Assume date is in dd/MM/yyyy format).

Create a Class **Main** that would get the HashMap as input and call the static method **retirementEmployeeList** present in the UserMainCode.

Input and Output Format:
The first line of the input consists of an integer n that corresponds to the number of employees.
The next 2 lines of the input consist of strings that correspond to the id and DOB of employee 1.
The next 2 lines of the input consist of strings that correspond to the id and DOB of employee 2.
and so on...
Output consists of the list of employee ids eligible for retirement in sorted order.

Sample Input :
4
C1010
02/11/1987
C2020
15/02/1980
C3030
14/12/1952
T4040
20/02/1950

Sample Output:
[C3030, T4040]

8. Kaprekar Number

Write a program to check whether the given input number is a Kaprekar number or not.
Note : A positive whole number “n” that has “d” number of digits is squared and split into two pieces: a right-hand piece that has “d” digits and a left-hand piece that has remaining “d” or “d-1” digits. If the sum of the two pieces is equal to the number, then “n” is a Kaprekar number.

The Kaprekar number assign to output variable 1, else -1.
Example 1:

Input 1:9
9^2 = 81, right-hand piece of 81 = 1 and left hand piece of 81 = 8
Sum = 1 + 8 = 9, i.e., equal to the number. Hence, 9 is a Kaprekar number.

Example 2:
Input 1:45
Hint:
45^2 = 2025, right-hand piece of 2025 = 25 and left hand piece of 2025 = 20
Sum = 25 + 20 = 45, i.e. equal to the number. Hence, 45 is a Kaprekar number."

Include a class **UserMainCode** with a static method "**getKaprekarNumber**" that accepts an integer argument and returns an integer. The method returns 1 if the input integer is a Kaprekar number, else the method returns -1.

Create a Class **Main** that would get the an integer as input and call the static method **getKaprekarNumber** present in the UserMainCode.

Input and Output Format:
Input consists of an integer.
Output consists of a single string that is either "Kaprekar Number" or "Not A Kaprekar Number."

Sample Input 1:
9

Sample Output 1:
Kaprekar Number

Sample Input 2:
45

Sample Output 2:
Kaprekar Number

Sample Input 3:
4

Sample Output 3:
Not A Kaprekar Number

9. Vowels

Based on a given string input, write a program to find the word that has the maximum number of vowels. If two or more words have the maximum number of vowels, print the first word.

Include a class **UserMainCode** with a static method "**storeMaxVowelWord**" that accepts a string argument and returns the word containing the maximum number of vowels.

Create a Class **Main** that would get the String as input and call the static method **storeMaxVowelWord** present in the UserMainCode.

Input and Output Format:
Input consists of a string. The string may contain both lower case and upper case letters.
Output consists of a string.

Sample Input :
What is your name?

Sample Output :
Your

10. ArrayList and Set Operations

Write a program that performs the following actions:

1. Read 2n integers as input and a set operator (of type char).
2. Create two arraylists to store n elements in each arraylist.
3. Write a function **performSetOperations** that accepts these two arraylists and the set operator as input.
4. The function would perform the following set operations:.

   "+" for SET-UNION

   "*" for SET-INTERSECTION

   "-" for SET-DIFFERENCE

   Refer to sample inputs for more details.
5. Return the final arraylist.

Include a class UserMainCode with the static method **performSetOperations** that accepts two arraylists and returns an arraylist.

Create a Class Main that would be used to read 2n+1 integers and call the static method present in UserMainCode.

Note:

- The index of first element is 0.

Input and Output Format:

Input consists of 2n+2 integers. The first integer denotes the size of the arraylist, the next "n" integers are values to the first arraylist, the next "n" integers are values to the second arraylist, and the last input corresponds to that set operation type.

Output consists of a modified arraylist as per step 4.

Refer sample output for formatting specifications.

Sample Input 1:

3

1

2

3

3

5

7

+

Sample Output 1:

1

2

3

5

7

Sample Input 2:

4

10

9

8

7

2

4

6

8

*

Sample Output 2:

8

Sample Input 3:

4

5

10

15

20

0

10

12

20

-

Sample Output 3:

5

15

11. max Scorer

Write a program that performs the following actions:

1. Read n strings as input and store them as arraylists. A string consists of student information like name and obtained scores in three subjects. Eg: name-mark1-mark2-mark3 [suresh-70-47-12]. The scores would range between 0 to 100 (inclusive).
2. Write a function **highestScorer** that accepts these arraylists and returns the name of the student who has scored the maximum marks. Assume the result will have only one student with maximum marks.

Include a class UserMainCode with the static method **highestScorer** that accepts the arraylist and returns the name (string) of highest scorer.

Create a Class Main that would be used to read n strings into arraylist and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 1 integer and n strings. The first integer denotes the size of the arraylist, and the next n strings are score patterns described above.

Output consists of a string with the name of the top scorer.

Refer sample output for formatting specifications.

Sample Input 1:

3

sunil-56-88-23

bindul-88-70-10

john-70-49-65

Sample Output 1:

john

12. Max Vowels

Write a program that fetches the word with maximum number of vowels. Your program should read a sentence as input from user and return the word with maximum number of vowels. In case there are two words of maximum length, it should return the word that comes first in the sentence.

Include a class UserMainCode with a static method **getWordWithMaximumVowels** that accepts a string. The return type is the longest word of type string.

Create a Class Main that would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with a maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

Appreciation is the best way to motivate

Sample Output 1:

Appreciation

13. Adjacent Swaps

Write a program that accepts a string as a parameter and returns the string with each pair of adjacent letters reversed. If the string has an odd number of letters, the last letter is unchanged.

Include a class UserMainCode with a static method **swapPairs** that accepts a string. The return type is a string with each pair of adjacent letters reversed.

Create a Class Main that would be used to accept two Input strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

forget

Sample Output 1:

ofgrte

Sample Input 2:

New York

Sample Output 2:

eN woYkr

14. Password

Based on a given string , write a program to find whether it is a valid password or not.

Validation Rule:
Atleast 8 characters
Atleast 1 number(1,2,3...)
Atleast 1 special character(@,#,%...)

Atleast 1 alphabet(a,B...)

Include a class **UserMainCode** with a static method "**validatePassword**" that accepts a string argument and returns a boolean value. The method returns true if the password is acceptable; otherwise, the method returns false.
Create a Class **Main** which would get a string as input and call the staticmethod **validatePassword** present in the UserMainCode.

Input and Output Format:
Input consists of a String.
Output consists of a String that is either "Valid" or "Invalid"

Sample Input 1:
hello@1010

Sample Output 1:
Valid

Sample Input 2:
punitha3

Sample Output 2:
Invalid

15. Employee Bonus

A Company wants to give bonuses to its employees. You have been assigned as the programmer to automate this process. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:

1. Read Employee details from the User. The details would include id, DOB (date of birth), and salary in the given order. The datatype for id, DOB, and salary is integer, string, and integer respectively.
2. You decide to build two HashMaps. The first HashMap contains employee id as key and DOB as value, and the second hashmap contains the same employee id as key and salary as value.
3. An employee in the age range of 25 to 30 years (inclusive) should get bonus of 20% of his salary and an employee in the range of 31 to 60 years (inclusive) should get a bonus of 30% of his salary. Store the result in TreeMap, with Employee ID as key and revised salary as value. Assume the age is calculated based on the date 01-09-2014. (Typecast the bonus to integer).
4. Other Rules:

   a. If Salary is less than 5000, store -100.

b. If the age is less than 25 or greater than 60, store -200.

c. “a” takes priority over “b,” i.e, if both a and b are true, then store -100.

5. You decide to write a function **calculateRevisedSalary** that takes the above HashMaps as input and returns the TreeMap as output. Include this function in class UserMainCode.

Create a Class Main that would be used to read employee details in step 1 and build the two HashMaps. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of employee details. The first number indicates the size of the group of employees. The next three values indicate the employee id, employee DOB, and employee salary. The Employee DOB format is “dd-mm-yyyy”

Output consists of a single string.

Refer sample output for formatting specifications.

Sample Input 1:

2

1010

20-12-1987

10000

2020

01-01-1985

14400

Sample Output 1:

1010

12000

2020

17280

16. Date Format

Write a program to read two String variables in DD-MM-YYYY. Compare the two dates and return the older date in 'MM/DD/YYYY' format.

Include a class UserMainCode with a static method **findOldDate** that accepts the string values. The return type is the string.
Create a Class Main that would be used to accept the two string values and call the static method present in UserMainCode.

Input and Output Format:
Input consists of two strings.
Output consists of a string.
Refer sample output for formatting specifications.

Sample Input 1:
05-12-1987
8-11-2010
Sample Output 1:
12/05/1987

17. Maximum Difference

Write a program to read an integer array and find the index of the larger number of the two adjacent numbers with largest difference. Print the index.
Include a class **UserMainCode** with a static method **findMaxDistance** that accepts an integer array and the number of elements in the array. The return type (Integer) should return index.

Create a Class Main that would be used to accept an integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers, where n corresponds the size of the array, followed by n integers.

Output consists of an Integer (index).

Refer sample output for formatting specifications.

Sample Input :
6
4
8
6
1
9
4

Sample Output :
4

[In the sequence 4 8 6 1 9 4, the maximum distance is 8 (between 1 and 9). The function should return the index of the greatest of two. In this case, it is 9 (which is at index 4) and output = 4.]

18. PAN Card

Write a program to read a string and validate PAN no. against following rules:
1. There must be eight characters.
2. First three letters must be alphabets, followed by four-digit number, and end with alphabet
3. All alphabets should be in upper case.

Print “Valid” if the PAN no. is valid; else, print “Invalid.”

Include a class **UserMainCode** with a static method **validatePAN** which accepts a string. The return type (Integer) should return 1 if the string is a valid PAN no.; else return 2.

Create a Class Main that would be used to accept a string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string, which corresponds to the PAN number.

Output consists of a string: "Valid" or "Invalid"

Refer sample output for formatting specifications.

Sample Input 1:

ALD3245E

Sample Output 1:

Valid

Sample Input 2:
OLE124F
Sample Output 2:
Invalid

19. Last Letters

Write a program to read a sentence as a string and store only the last letter of each word of the sentence in upper case separated by $. Print the final string.

Include a class **UserMainCode** with a static method **getLastLetter** that accepts a string. The return type (string) should return the final string.
Create a Class Main that would be used to read a string, and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string (the final string).

Refer sample output for formatting specifications.

Smaple Input :
This is a cat
Sample Output :
SSA$T

20. Largest Key in HashMap

Write a program that constructs a HashMap and returns the value corresponding to the largest key.

Include a class UserMainCode with a static method **getMaxKeyValue** that accepts a string. The return type (String) should be the value corresponding to the largest key.

Create a Class Main that would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the HashMap. The next n pair of numbers equals the integer key and value as string.

Output consists of a string, which is the value of largest key.

Refer sample output for formatting specifications.

Sample Input 1:

3

12

amron

9

Exide

7

SF

Sample Output 1:

amron

21. Day of the Week

Write a program to read a date as string (MM-dd-yyyy) and return the day of week on that date.

Include a class UserMainCode with a static method **getDay** that accepts the string. The return type (string) should be the day of the week.

Create a Class Main that would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

07-13-2012

Sample Output 1:

Friday

22. Transfer from HashMap to Arraylist

Write a program that constructs a HashMap with “employee id” as key and “name” as its value. Based on whether the rules below are satisfied, the name must be added to the arraylist.

i)First character should be in lower case and the last character should be in upper case.

ii)In name, at least one digit should be present.

Include a class UserMainCode with a static method **getName** that accepts a HashMap. The return type is an arraylist as expected in the above statement.

Create a Class Main that would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the HashMap. The next n pair of numbers contains the employee id and name.

Output consists of arraylist of strings as mentioned in the problem statement.

Refer the sample output for formatting specifications.

Sample Input 1:

4

1

ravi5raJ

2

sita8gitA

3

ram8sitA

4

rahul

Sample Output 1:

ravi5raJ

sita8gitA

ram8sitA

23. Date Format Conversion

Consider a date string in the format dd/mm/yyyy. Write a program to convert the given date to the format dd-mm-yy.

Include a class **UserMainCode** with a static method "**convertDateFormat**" that accepts a String and returns a String.

Create a Class **Main** that would get a String as input and call the static method **convertDateFormat** present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input:

12/11/1998

Sample Output:

12-11-98

24. String Processing - ZigZag

Write a program to read a string containing date in DD-MM-YYYY format. Find the number of days in the given month.

Note: In leap year, February has 29 days.

Include a class UserMainCode with a static method **getLastDayOfMonth** that accepts the string. The return type is the integer that has number of days.

Create a Class Main that would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:
Input consists of a string.
Output consists of an integer.
Refer sample output for formatting specifications.

Sample Input 1:
12-06-2012
Sample Output 1:
30

Sample Input 2:
10-02-2012
Sample Output 2:
29

25. Age for Voting

Given a date of birth (dd/MM/yyyy) of a person in string, compute his age as of 01/01/2015.

If his age is greater than 18, then print “eligible,” else print “not-eligible.”

Include a class UserMainCode with a static method getAge that accepts the string value. The return type is the string.

Create a Class Main that would be used to accept the two string values and call the static method present in UserMainCode.

Input and Output Format:
Input consists of two strings.
Output consists of a string.
Refer sample output for formatting specifications.

Sample Input 1:
16/11/1991

Sample Output 1:
eligible

26. Constructor Overloading

Create a class named Product with the following private member variables.

- id of type Long
- productName of type String
- supplierName of type String

Include appropriate getters and setters.

Include a 3-argument constructor, 2-argument constructor, and a default constructor. Assume that most of the products are supplied by “Nivas” suppliers. In the 2-argument constructor, set the value of supplierName to “Nivas.”

Include a method named display. It does not accept any arguments, and its return type is void. Display the details of the product in this method. The method prototype is void display();

Create another class and write a main method to test the above class.

Input and Output Format:

Refer the sample input and output for formatting specifications.

All text in bold corresponds to input and the rest corresponds to output.

Sample Input and Output 1:

Enter the product id

1

Enter the product name

Printer

Is the product supplied by Nivas Suppliers? Type “yes” or “no” (not case sensitive)

Yes

Product Id is 1

Product Name is Printer

Supplier Name is Nivas

Sample Input and Output 2:

Enter the product id

1

Enter the product name

Printer

Is the product supplied by Nivas Suppliers? Type yes or no (not case sensitive)

NO

Enter the supplier name

HP

Product Id is 1

Product Name is Printer

Supplier Name is HP

27. Book

Create a class named Author with the following private member variables.

- name of type String
- email of type String
- gender of type String

The class Author implements Comparable interface.

Include appropriate getters and setters.

Include appropriate constructors.

Create a class named Book with the following private member variables

- private String name;
- private List<Author> authorList;
- private double price;
- private int qtyInStock = 0; // Default value --- 0

Include appropriate getters.

Include default constructor, 4-argument constructor, and a 3-argument constructor (qtyInStock is set to 0).

Override the toString() method defined in the Object class to display the book details in the format as shown in the sample output.

Input and Output Format:

Refer the sample input and output for formatting specifications.

All text in bold corresponds to input and the rest corresponds to output.

Sample Input and Output 1:

Enter the book name

Fundamentals of Algms

Enter the number of authors

3

Enter the author name

Horowitz

Enter the author email id

hor@gmail.com

Enter the author's gender

Male

Enter the author name

Sahni

Enter the author email id

sahni@gmail.com

Enter the author's gender

Male

Enter the author name

Rajasekaran

Enter the author email id

raja@gmail.com

Enter the author's gender

Male

Enter the book price

560

Is the book currently available? Type “Yes/No” (Not case sensitive)

No

Fundamentals of Algms authored by Horowitz Sahni Rajasekaran costs Rs.560.0: Not Available

Sample Input and Output 2:

Enter the book name

Fundamentals of Algms

Enter the number of authors

3

Enter the author name

Horowitz

Enter the author email id

hor@gmail.com

Enter the author's gender

Male

Enter the author name

Sahni

Enter the author email id

sahni@gmail.com

Enter the author's gender

Male

Enter the author name

Rajasekaran

Enter the author email id

raja@gmail.com

Enter the author's gender

Male

Enter the book price

560

Is the book currently available? Type Yes/No (Not case sensitive)

Yes

Enter the number of books available

5

Fundamentals of Algms authored by Horowitz Sahni Rajasekaran costs Rs.560.0 : Available

28. Employee Register

You are given the responsibility to collect and store all the employee details in your organization. Until now, all these details were collected in employee register notebook. Since this involves a lot of bookkeeping, you want to change the system to a computer-based application.

You are trained in Core Java/C# training, so you decided to design an object-oriented system. As a first step, you want to get employee information such as
First Name
Last Name
Mobile Number
Email Address

There is no higher limit to store the number of employee in the application. Get the number

of employees first before getting all the employee details.

All the employee details are collected and stored in ArrayList. For most of the reports, you want to order the employees based on their first name. So, once you collect all the employee details, print all of them sorted by their first name.

Note:
In case of Java, Use Java Format Specifier:

System.out.format("%-15s %-15s %-15s %-30s %-15s\n","Firstname","Lastname","Mobile","Email","Address");

Sample Output:

Enter The Number of Employees

2

Enter Employee 1 Details:

Enter the Firstname

William

Enter the Lastname

Becker

Enter the Mobile

449876543210

Enter the Email

william@abc.com

Enter the Address

America

Enter Employee 2 Details:

Enter the Firstname

Amar

Enter the Lastname

CR

Enter the Mobile

9874561230

Enter the Email

amar@gmail.com

Enter the Address

London

Employee List:

Firstname	Lastname	Mobile	Email	Address
Amar	CR	9874561230	amar@gmail.com	London
William	Becker	449876543210	william@abc.com	America

29. Collect unique symbols from a set of cards

Playing cards during travel is a fun-filled experience. For this game, users wanted to collect all four unique symbols. Can you help these guys to collect unique symbols from a set of cards?

Create Card class with attributes symbol and number. From the main method, collect details of each card (symbol and number) from the user.
Collect all these cards in a set, since set is used to store unique values or objects.
Cards need to be compared with each other to identify whether both the cards are of same symbol. For this, we need to implement equals method.

Once we collect all four different symbols, display the first occurrence of card details in alphabetical order.

Sample input output

Enter a card :
a
1
Enter a card :
a
2
Enter a card :
a
7
Enter a card :
d
6
Enter a card :
c
2
Enter a card :
d
1
Enter a card :

c
1
Enter a card :
b
2
Four symbols gathered in 8 cards.
Cards in Set are :
a 1
b 2
c 2
d 6

30. Set of boxes

Problem Statement :

Write a Program to insert the box details into the set.

Problem Constraints :

1. Create a Box Class with attributes length(Double), width(Double), and height(Double).
2. Get the Box details from the user and insert them to the Set.
3. We need to store the details of boxes with different volumes. When the volume of a new box is the same as the volumes of previous boxes included in the Set, don't insert this box in the Set.
4. The Boxes are said to be the same when their volumes are equal.
5. Override the equals method to compare the box volumes.

Sample Input and Output :

Enter the number of Box
5
Enter the Box 1 details
Enter Length
2.1
Enter Width
1.2
Enter Height
2.1
Enter the Box 2 details
Enter Length
3.2
Enter Width
2.3
Enter Height
3.2
Enter the Box 3 details
Enter Length

1.2
Enter Width
2.1
Enter Height
1.2
Enter the Box 4 details
Enter Length
3.2
Enter Width
2.3
Enter Height
3.2
Enter the Box 5 details
Enter Length
3.3
Enter Width
2.2
Enter Height
1.1
Unique Boxes in the Set are
Length =1.2 Width =2.1 Height =1.2 Volume =3.02
Length =2.1 Width =1.2 Height =2.1 Volume =5.29
Length =3.3 Width =2.2 Height =1.1 Volume =7.99
Length =3.2 Width =2.3 Height =3.2 Volume =23.55

31. Profit or Loss

Sam purchased x dozens of toys at the rate of Rs. y per dozen. He then sold each toy at the rate of Rs. z. Help him calculate his percentage profit.

Given the values of x, y, and z, write a program to compute Sam's profit percentage.

Hint:

For x=20, y=375 and z=33

Cost Price of 1 toy = 375/12 = Rs. 31.25

Selling Price of 1 toy = Rs.33

So gain = 33-31.25 = Rs.1.75

Profit % = 1.75/31.25*100 = 5.6%

Input Format:

Input consists of 3 integers: x, y, and z. x is number of dozens purchased. y is cost per dozen. z is selling price per item.

Output Format:

Refer the Sample Input and Output for formatting details. The profit percentage needs to be printed correct to 2 decimal places.

Sample Input and Output:

[All text in bold corresponds to input and the rest corresponds to output]

Enter the number of dozens of toys purchased

20

Enter the price per dozen

375

Enter the selling price of 1 toy

33

Sam's profit percentage is 5.60 percent

32. Math class

This program is to illustrate the difference between static methods and non-static methods.

Write a program that accepts 2 “Integer” class type values as input, prints their absolute values, and checks their equality. Use the abs() method defined in the Math class and the equals() method defined in the Object class.

Input and Output Format:

Refer the sample input and output for formatting specifications.
All text in bold corresponds to input and the rest corresponds to output.

Sample Input and Output 1:

Enter the first integer
4
Enter the second integer
4
Absolute value of 4 is 4
Absolute value of 4 is 4
4 = 4

Sample Input and Output 2:

Enter the first integer
-5
Enter the second integer
6
Absolute value of -5 is 5
Absolute value of 6 is 6
-5 != 6

33. Wrapper class

This program is to illustrate the parseInt() method defined in the Integer Wrapper class.

Write a program that accepts 3 String values as input and invokes some of the methods defined in the Integer Wrapper class.

Refer to the sample input and output. All functions should be performed using the methods defined in the Integer class.

Input and Output Format:

Refer the sample input and output for formatting specifications.
All text in bold corresponds to input, and the rest corresponds to output.

Sample Input and Output :
Enter the binary number
111
Enter the octal number
11
Enter the hexadecimal number
1F
The integer value of the binary number 111 is 7
The integer value of the octal number 11 is 9
The integer value of the hexadecimal number 1F is 31

34. Operations on String List

Problem Statement:
Write a program to perform basic operations like insert, delete, display, and search in list. List contains string object items where these operations are to be performed.

Sample Input and Output 1:
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :

1
Enter the item to be inserted:
Bottle
Inserted successfully
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
1
Enter the item to be inserted:
Water
Inserted successfully
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
1
Enter the item to be inserted:
Cap
Inserted successfully
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
1
Enter the item to be inserted:
Monitor
Inserted successfully
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
2
Enter the item to search :
Mouse
Item not found in the list.
1. Insert
2. Search

3. Delete
4. Display
5. Exit
Enter your choice :
2
Enter the item to search :
Monitor
Item found in the list.
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
3
Enter the item to delete :
Mouse
Item does not exist.
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
4
The Items in the list are :
Bottle
Water
Cap
Monitor
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
3
Enter the item to delete :
Cap
Deleted successfully
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :

4
The Items in the list are :
Bottle
Water
Monitor
1. Insert
2. Search
3. Delete
4. Display
5. Exit
Enter your choice :
5

35. Collect and group cards

Write a program to collect and store all the cards to assist the users in finding all the cards in a given symbol.
This card game consist of N number of cards. Get the details of N number of card from the user and store the values in Card object with the attributes symbol and number.
Store all the cards in a map with symbol as its key and list of cards as its value. Map is used here to easily group all the cards based on their symbol.

Once all the details are captured, print all the distinct symbols in alphabetical order from the Map. For each symbol, print all the card details, number of cards, and their sum respectively.

Sample input output :

Enter Number of Cards :
13
Enter card 1:
s
1
Enter card 2:
s
12
Enter card 3:
s
13
Enter card 4:
d
4
Enter card 5:
c
5
Enter card 6:
h

5
Enter card 7:
h
7
Enter card 8:
c
3
Enter card 9:
c
2
Enter card 10:
h
9
Enter card 11:
s
7
Enter card 12:
d
4
Enter card 13:
d
3
Distinct Symbols are :
c d h s
Cards in c Symbol
c 5
c 3
c 2
Number of cards : 3
Sum of Numbers : 10
Cards in d Symbol
d 4
d 4
d 3
Number of cards : 3
Sum of Numbers : 11
Cards in h Symbol
h 5
h 7
h 9
Number of cards : 3
Sum of Numbers : 21
Cards in s Symbol
s 1
s 12
s 13

s 7
Number of cards : 4
Sum of Numbers : 33