**Question 1: Profile Picture**   
  
     Roy wants to change his profile picture on Facebook. Now Facebook has some restriction over the dimension of picture that we can upload.Minimum dimension of the picture can be L x L, where L is the length of the side of square.  
  
Dimension of a photo is denoted as W x H  
where W - width of the photo and H - Height of the photo  
  
When any photo is uploaded following events may occur:  
  
[1] If any of the width or height is less than L, user is prompted to upload another one. Print "UPLOAD ANOTHER" in this case.  
[2] If width and height both are larger than length L , user is prompted to crop it. Print "CROP IT" in this case.  
[3]   If the photo is already square and width and height  is equal to L ,  then it is accepted. Print "ACCEPTED" in this case.  
(quotes are only for clarification)  
  
Given L, W and H as input, Write a program to print appropriate text as output.  
  
**Input Format:**  
First input is an integer that denotes  L , length of the square.  
Second line consists of two space separated integers W and H that denotes the width and height of the photo.  
  
**Output Format:**  
Output is appropriate string for the inputs ,  UPLOAD ANOTHER/CROP IT/ACCEPTED.  
  
**Sample Input 1:**  
180  
640 480  
  
**Sample Output 1:**  
CROP IT  
  
**Sample Input 2:**  
180  
120 300  
  
**Sample Output 2:**  
UPLOAD ANOTHER

**Question 2:** **Mileage Calculation**  
  
    Raju is fond of bikes.He convinced his parents and bought a new bike.Now he wants to check the mileage of his bike.He is weak in mathematics,so help him to find out the mileage.Petrol price(X),Distance covered in kilometres(Y) and Amount(Z) for which he has filled the petrol is given as inputs.Write a program to find out the mileage of the bike.  
  
**Input Format:**  
First input is a double value that corresponds to the X value.  
Second input is a double value that corresponds to the Y value.  
Third input is a integer value that corresponds to the Z value.  
  
**Output Format:**  
Output is a double value that denotes the mileage of the bike(rounded off to two decimal places).  
      
**Sample Input 1:**  
70.77  
280.50  
500  
  
**Sample Output 1:**  
39.70  
  
**Sample Input 2:**  
55.90  
320.25  
400  
  
**Sample Output 2:**  
44.75

**Question 3: MobileBrand - Requirement 1**

**Requirement 1:**

Let’s start off by creating two **Mobile** objects and check whether they are equal.

1. Create a **Mobile** Class with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| referenceId | String |
| modelName | String |
| displaySize | Double |
| Price | Double |
| launchedDate | java.util.Date |

1. Mark all the attributes as private
2. Create / Generate appropriate Getters & Setters
3. Add a default constructor and a parameterized constructor to take in all attributes in the given order:   
   **Mobile ( String referenceId, String modelName, Double displaySize, Double price, java.util.Date launchedDate )**
4. When the “Mobile” object is printed, it should display the following details: **[Override the toString method]**  
   Print format:  
   Reference Id: "referenceId"  
   Model Name: "modelName"  
   Display Size: "displaySize"  
   Price: "price"  
   Launched Date: "launchedDate"
5. Two Mobiles are considered same if they have the same referenceId, and modelName. Implement the logic in the appropriate function. (Case – Insensitive) **[Override the equals method]**  
     
   The input format consists of Mobile details separated by a comma in the below order,  
   referenceId, modelName, displaySize, price, launchedDate

The Input to your program would be details of two Mobiles, you need to display their details as given in "5th point(refer above)" and compare the two Mobiles and display if the Mobiles are same or different.  
  
**Note:**There is an empty line between display statements. Print the empty lines in the main function.  
              Display one digit after the decimal point for the Double data type.  
  
  
**Sample Input and Output 1:**

Enter mobile 1 detail:

**#SM 45 JJ6-001,Galaxy J6,5.6,13990,02-01-2017**

Enter mobile 2 detail:

**#SM 45 JJ6-001,Galaxy J6,5.6,13990,02-01-2017**

Mobile 1

Reference Id: #SM 45 JJ6-001

Model Name: Galaxy J6

Display Size: 5.6

Price: 13990.0

Launched Date: 02-01-2017

Mobile 2

Reference Id: #SM 45 JJ6-001

Model Name: Galaxy J6

Display Size: 5.6

Price: 13990.0

Launched Date: 02-01-2017

Mobile 1 is same as Mobile 2

**Sample Input and Output 2:**

Enter mobile 1 detail:

**#SM 45 JJ6-001,Galaxy J6,5.6,13990,02-01-2017**

Enter mobile 2 detail:

**#SM 45 JJ6-001,Galaxy J7,5.6,13990,02-01-2017**

Mobile 1

Reference Id: #SM 45 JJ6-001

Model Name: Galaxy J6

Display Size: 5.6

Price: 13990.0

Launched Date: 02-01-2017

Mobile 2

Reference Id: #SM 45 JJ6-001

Model Name: Galaxy J7

Display Size: 5.6

Price: 13990.0

Launched Date: 02-01-2017

Mobile 1 and Mobile 2 are different

**Question 4: MobileBrand - Requirement 2**

**Requirement 2:**  
Now we are gonna start creating a MobileBrand and add Mobiles to it. Start with creating a MobileBrand and use menu-driven approach to add, remove, display details of the Mobiles in the MobileBrand.  
  
a)Create a Class **Mobile** with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| referenceId | String |
| modelName | String |
| displaySize | Double |
| price | Double |
| launchedDate | java.util.Date |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a default constructor and a parameterized constructor to take in all attributes in the given order:  **Mobile( String referenceId, String modelName, Double displaySize, Double price, java.util.Date launchedDate )**  
  
b)Create a Class **MobileBrand** with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| name | String |
| mobileList | List<Mobile> |

Mark all the attributes as private, Create / Generate appropriate Getters & Setters, Add a default constructor and a parameterized constructor to take in all attributes in the given order: **MobileBrand( String name, List<Mobile> mobileList ).**In constructor pass the mobileList value as an empty list. Only one MobileBrand will be present at a time.  
  
c) Create the following static method in Mobile class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public static Mobile createMobile(String detail) | This method accepts a string which contains mobile details separated by commas. Split the details and create a mobile object from the details and return it. |

The mobile details should be given as a comma-separated value in the below order,  
**referenceId, modelName, displaySize, price, launchedDate**  
  
d) Create the following methods in MobileBrand class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public void addMobileToMobileBrand(Mobile mobile) | This method accepts a mobile object and adds the mobile to the mobile list of the current MobileBrand. |
| public Boolean removeMobileFromMobileBrand(String referenceId) | This method will get the referenceId of the mobile and delete the mobile with the specified referenceId from the current MobileBrand. If a mobile with the given referenceId found, delete the mobile and return**true**. If a mobile with the referenceId is not found return **false**. |
| public void displayMobiles() | This method will display the mobile list in the current MobileBrand. If the mobile list is empty display "**No mobiles to show"**, else display "Mobiles in [mobileBrand name]" and display all the mobile details in the specified format. Where [mobileBrand name] specifies the name of the mobileBrand. |

After deletion, if true is returned print "**Mobile successfully deleted**", else print "**Mobile not found in the mobileBrand**".  
  
**Note:** The above print statements should be present in the main method.  
  
Print format: **System.out.format("%-15s %-15s %-12s %-8s %s\n","Reference Id","Model Name","Display Size", "Price","Launched Date");**  
  
**Display 1 digit after decimal point in Double.**  
  
**Sample Input and Output:**

Enter the name of the Mobile Brand:

**Samsung**

1.Add Mobile

2.Delete Mobile

3.Display Mobiles

4.Exit

Enter your choice:

**1**

**#SM 45 JJ6-001,Galaxy J6,5.6,13990,02-01-2017**

Mobile successfully added

1.Add Mobile

2.Delete Mobile

3.Display Mobiles

4.Exit

Enter your choice:

**3**

Mobiles in Samsung

Reference Id    Model Name      Display Size Price    Launched Date

#SM 45 JJ6-001  Galaxy J6       5.6          13990.0  02-01-2017

1.Add Mobile

2.Delete Mobile

3.Display Mobiles

4.Exit

Enter your choice:

**2**

Enter the reference id of the mobile to be deleted:

**#SM 45 JJ6-001**

Mobile successfully deleted

1.Add Mobile

2.Delete Mobile

3.Display Mobiles

4.Exit

Enter your choice:

**2**

Enter the reference id of the mobile to be deleted:

**#SM 45 JJ6-001**

Mobile not found in the Mobile Brand

1.Add Mobile

2.Delete Mobile

3.Display Mobiles

4.Exit

Enter your choice:

**3**

No mobiles to show

1.Add Mobile

2.Delete Mobile

3.Display Mobiles

4.Exit

Enter your choice:

**4**

**Question 5: Mobile Brand Requirement 3**

**Requirement 3:**  
   In this requirement, you need to validate the referenceId of the Mobile.  
  
a)Create a class **Main** with the following static methods:

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| static Boolean validateReferenceId(String referenceId) | Validate the referenceId based on the rules given below. Returns**true**ifreferenceId is valid else return **false** |

b) While validating a referenceId follow the below rules.  
  
1. The first part should start with '#' and then followed 2 to 3 UPPERCASE letters.  
2. The second part contains a two digit number. Sometimes second part will be neglated.  
3. Third part contains 2 to 6 UPPERCASE letters and numbers followed by a '-' and 2 to 4 digit number.  
4. Each part is separated by space.

Valid Format**:#BA 45 CX-20**  
                     **#BA CX-20**  
                               **#JIO PH201-20**  
  
**Note:** Print "**Reference Id is valid**" if referenceId is valid else print "**Reference Id is invalid**".  
            All the above print statements are present in the main method.  
  
**[All text in bold corresponds to input]**  
**Sample Input and Output 1:**  
  
Enter the reference Id to be validated:  
**#RM 87 AAA-062**  
Reference Id is valid  
  
**Sample Input and Output 2:**  
  
Enter the reference Id to be validated:  
**#MT 57 X-067**  
Reference Id is invalid