Social Media Task – Get ready for some fun

Social Media

- Create an abstract class SocialMedia

- The SocialMedia class will have the following fields:

  - Personal url (String)
  - Account length (int)
  - Platform (static String)
  - Username
  - Password
  - Full name

- The SocialMedia class will have the following abstract methods:

  - Direct messaging (String username, String message)
  - Post (Object media)   [using Object type as the parameter]
  - Notifications ()

Post class

- Create a class that has the following instance variable:
  String body, and String dateTime

  - Encapsulate body. Provide a public getter and setter
  - Make dateTime final, private, read only (getter, but no setter)

- Create a constructor that will take and initialize the body instance variable. Upon creation of the post the current date and time should be taken and stored into the dateTime variable. Use the code given below

  Note: Since we didn't learn this class use the follow code:

this.dateTime = LocalDateTime.now().format(DateTimeFormatter.ofPattern("MMM dd, yyyy | hh:mm a"));

➔ Need to have these two imports for the above code:

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
```

Facebook

- Create a class for a Facebook which inherits the SocialMedia class and has additional instance variables:
  age, number of friends, and posts (ArrayList of Post)

- Encapsulate all the variables.

- Set the platform for "Facebook" using static block

- Create a constructor that will create a Facebook by taking the username and password.

- If the password contains the username, then it is not a valid password and should not be saved. In this case Print “Password contained username. Default password created: ‘password’ and set the password for the user to the default of ‘password’.

- Create and assign the user's person url by using 'facebook.com/' and adding their username at the end

- Create an empty ArrayList of Posts

- Overload the constructor to accept the username, password, and the user’s name. Chain previous constructor

- If the name entered has any characters that are not letters from the alphabet, then the name is invalid. In this case print “invalid name” and assign the name value to be ‘no name’

- Overload the constructor to accept the username, password, the user’s name, age and number of friends. Chain previous constructor

- Only store the age and number of friends into the variables if they are valid numbers.

- The age cannot be a negative number
- The number of friends cannot be negative or more than 5000.

If they are invalid print “Invalid (age/number of friends) and keep the values as the default of 0.

- Implement all methods coming from Social Media class

    - Direct messaging (String username, String message)
      - print = %message sent to %username

    - Post (Object body)
      - Creates a Post object (the class we made earlier, and store the Post objects to the ArrayList of Posts)

    - Notifications ()
      - Checks the current time. If the time is between 8 am - 5 pm print "Notification", otherwise print "Sleep mode"

Note: use this code to get the current hour: LocalTime.now().getHour()

- The return value will be given in 24 hour format.

- Override the toString method to print all the information of a Facebook

- Create a sendFriendRequest method that will accept a Facebook object and send them a friend request. The method should return a boolean:

  if the friend request was sent (true) or if the friend request was not sent (false).

  - If your friend list is at the 5000 limit then do not send the request and print “You have reached the limit of friends.”

- If the user you are trying to send the request to has already hit the limit, then print “$theUsersName cannot accept any more friend request.

- If both the user and you are under the limit print “Friend request sent to $theUsersName” and increase your number of friends by one.

Groups interface

- Create an interface hasGroups with the following actions:

  - joinGroup(String group);
  - leaveGroup(String group);

- Add a variable in Facebook to count the number of groups they are in and implement the 2 methods from Groups interface.

  - joinGroup prints "%usersName has joined %group" and increment the total number of groups
  - leaveGroup prints "%usersName has left %group" and decrement the total number of groups

---

Picture class

- Create a class that has the following instance variable:
  - byte [] image
  - extension (String)
  - caption

  - Encapsulate the caption.
  - make the image and extension final, private, and read-only (getter, but no setter)

- Create a constructor that will take and initialize image, extension, and caption

---

Instagram Class:

- Create a class Instagram that will inherit the SocialMedia class

- Add additional instance variables: number of followers, number of followings, and pictures (ArrayList of Pictures)

- Set the platform for all objects to be "Instagram"

- Encapsulate the variables

- Create a constructor to initialize the username and password
  - Set up the user's person url:
    "Instagram.com/" + $username

- Implement all the abstract methods coming from the Social Media class

  - Direct messaging (String username, String message)

  - print = %message sent to %username

  - Post (Object body)

  - Let’s keep it simple and say our variable body will be a picture)
    Add the picture object

  - Notifications ()

- Checks the current time. If the time is between 10 am - 8 pm print "Notification", otherwise print "Sleep mode"

Note: use this code to get the current hour: LocalTime.now().getHour()

- The return value will be given in 24 hour format.

Pictures Interface

- Create hasPictures interface and declare the following actions:

  - likePicture()
  - unLikePicture()
  - sharePicture()

- Implement this interface to the Instagram class. (The actual implementation does not matter. Print statement related to action. Focus on idea of how abstraction is working)

Extra: Make a bunch of objects and try different things out

- Create an ArrayList of Social Media objects to store both Facebook and Instagram objects

- Find all the Facebook objects that had less than 5000 friends to they could be added

- Go through all the Instagram objects and like a picture from each object

- Check which user has both Instagram and Facebook object by comparing their full names