День 13

Проработка тем прошедших дней

Доп. статья:

- Класс `java.util.Date` и работа с ним: https://javarush.ru/groups/posts/1941-kak-ne-poterjatjhsja-vo-vremeni--datetime-i-calendar

Задача:

Нам необходимо создать свою небольшую социальную сеть на Java.

Для этого мы должны реализовать следующие классы:
`User` - сущность “Пользователь”
`Message` - сущность “Сообщение”
`MessageDatabase` - класс, который будет заниматься хранением сообщений
`Test` - класс, где в методе `main()` мы будем тестировать нашу соц. сеть

Класс `User` должен иметь следующую структуру:

Поля:

- Строковое поле “имя пользователя” (англ. `username`)
- Список, параметризованный классом `User`, с названием “подписки” (англ. `subscriptions`). В этом списке должны храниться те пользователи, на которых подписан пользователь.

Конструктор:
Должен принимать в качестве аргумента только имя пользователя. Также, должен инициализировать поле “подписки” пустым списком.

Методы:

- Геттеры на все поля

- `public void subscribe(User user)` - подписывает пользователя на пользователя `user`

- `public boolean isSubscribed(User user)` - возвращает `True`, если пользователь подписан на пользователя `user` и `False`, если не подписан.

- `public boolean isFriend(User user)` - возвращает `True`, если пользователь `user` является другом и `False`, если пользователь `user` не является другом. Подумайте, что такое дружба в контексте соц. сетей.

- `public void sendMessage(User user, String text)` - отправляет сообщение с текстом `text` пользователю `user`. Здесь должен использоваться статический метод из `MessageDatabase`.

- `public String toString()` - возвращает строковое представление пользователя (имя пользователя).

Класс `Message` должен иметь следующую структуру:

Поля:
- Поле типа `User` “отправитель” (англ. `sender`) - отправитель сообщения
- Поле типа `User` “получатель” (англ. `receiver`) - получатель сообщения
- Строковое поле “текст” (англ. `text`) - текст сообщения
- Поле типа `Date` “дата” (англ. `date`) - дата отправки сообщения

Конструктор:
- Конструктор должен принимать 3 аргумента - отправителя, получателя и текст сообщения. Также, конструктор должен назначать полю `date` текущее время (то есть время создания объекта `Message`).

Методы:
- Геттеры на все поля
- Метод `toString()`, который возвращает строковое представление сообщения в таком формате:

```
FROM: 'Имя отправителя'
TO: 'Имя получателя'
ON: Sun Aug 30 19:07:34 MSK 2020
'Текст сообщения'
```

*в поле `ON` должна быть дата создания объекта класса `Message`

Класс `MessageDatabase` должен иметь следующую структуру:

Поля:
- Статическое поле “сообщения” (англ. `messages`), которое будет хранить список из сообщений (объектов класса `Message`). Это поле должно инициализироваться пустым списком. Этот список и есть наша условная “база данных”, которая хранит все сообщения в соц. сети.

Конструктор:
Нет конструктора. Объекты класса `MessageDatabase` создаваться не будут (все методы и поля статические).

Методы:

- `public static void sendMessage(User u1, User u2, String text)` - этот метод “отправляет” новое сообщение от пользователя `u1` пользователю `u2` с текстом `text`. Отправка в нашем контексте означает добавление сообщения в нашу “базу данных”.
- `public static List<Message> getMessages()` - возвращает список всех сообщений в “базе данных”.
- `public static void showDialog(User u1, User u2)` - этот метод должен вывести цепочку сообщений (диалог) пользователей `u1` и `u2`. Формат вывода должен быть таким:

```
user1: Привет!
user2: Привет, user1!
user1: Как у тебя дела?
user2: Все ок, спасибо :)
```

* У вас могут быть любые другие сообщения. Где `user1` и `user2` - имена пользователей `u1` и `u2`.

Класс `Task1`:

В методе `main()` этого класса необходимо создать трех пользователей.
Затем необходимо:

- Отправить 2 сообщения от пользователя 1 пользователю 2
- Отправить 3 сообщения от пользователя 2 пользователю 1
- Отправить 3 сообщения от пользователя 3 пользователю 1
- Отправить 3 сообщения от пользователя 1 пользователю 3
- Отправить 1 сообщение от пользователя 3 пользователю 1

Сообщения могут быть любыми.
После этого, необходимо вывести диалог пользователя 1 и пользователя 3 с помощью метода `showDialog()`.