EECS 293

Software Craftsmanship

2014 Fall Semester

Programming Assignment 3

Due at your recitation session on September 15-19, 2014

Reading

Read Chapters 16, 19.5, and 19.6.

Programming

In this programming assignment, you will revisit the previous programming assignment (`SocialNetwork`) to accomplish two additional objectives: reduce the McCabe complexity of your software and its use of non-structured programming.

Additionally, make the following changes to the `SocialNetwork`.

The class `User` should contain the first, middle, and last name of the user, his email and phone number. These fields are optional and by default are null. They can be set and retrieved with:

- `User setFirstName(String name)` sets (or replaces) the first name of this `User`, returning this `User`. If the identifier is null, throw a `NullPointerException`. If the `User` is invalid, throw a `UninitializedObjectException`.
- `String getFirstName()` returns the first name of this `User`.

Similar methods should be provided for:

- Middle name
- Last name
- Email
- Phone number

(You should also think about the reasons for the return type of `setFirstName`.)

It is unlikely but possible that `User.toString` would return the `String` "Invalid User: Uninitialized ID" even for a valid `User`. Implement a check so that this string is returned only for invalid users.

The class `Link` should return better information when there is a failure in establishment and tear down. A common error handling technique is to use a status variable (for example, Section 17.3: "Rewrite with a status variable"). Create an enumerated type called `SocialNetworkStatus` whose possible values are `SUCCESS`, `ALREADY_VALID`, `INVALID_USERS`, `INVALID_DATE`, `ALREADY_ACTIVE`, `ALREADY_INACTIVE`. Then, methods can have a `Status` argument that is set during the method execution so that on return the `Status` signifies the method's outcome. Refactor the following `Link` methods:

- `void setUsers(Set<User> users, SocialNetworkStatus status)` set this `Link` to connect the two users and marks the link as valid.
    - If this link had already been associated with two users (that is, if the link was already valid), make no changes to this `Link` and set status to `ALREADY_VALID`.
    - If the users argument is anything but two distinct users, make no changes to this `Link` and set status to `INVALID_USERS`.
- `void establish(Date date, SocialNetworkStatus status)` establish the link at the given date.
    - If this `Link` is invalid, throw a `UninitializedObjectException`.
    - If this `Link` was already active, make no changes to this `Link` and set the status to `ALREADY_ACTIVE`.
    - If the given date precedes the last date on record, make no changes to this `Link` and set the status to `INVALID_DATE`.
- `void tearDown(Date date, SocialNetworkStatus status)` tear down the link at the given date.
    - If this `Link` is invalid, throw a `UninitializedObjectException`.
    - If this `Link` was already inactive, make no changes to this `Link` and set the status to `ALREADY_INACTIVE`.
    - If the given date precedes the last date on record, make no changes to this `Link` and set status to `INVALID_DATE`.

These methods:

- Throw a `NullPointerException` if any argument is null.
- In case of success, set the status `SUCCESS`.

Refactor similarly the class `SocialNetwork`:

- `void establishLink(Set<String> ids, Date date, SocialNetworkStatus status)` establish a link between the two users with the given identifiers at the given date.
    - If ids is anything but two distinct users that are already present in this `SocialNetwork`, set the status to `INVALID_USERS`.
    - If the link was already active, make no changes to the `SocialNetwork` and set the status to `ALREADY_ACTIVE`.

    - If the given date precedes the last date on record, make no changes to the `SocialNetwork` and set the status to `INVALID_DATE`.
- `void tearDownLink(Set<String> ids, Date date, SocialNetworkStatus status)` tear down the link between the two users with the given identifiers at the given date.
    - If ids is anything but two distinct users that are already present in this `SocialNetwork`, set the status to `INVALID_USERS`.
    - If the link was already inactive, make no changes to the `SocialNetwork` and set the status to `ALREADY_INACTIVE`.
    - If the given date precedes the last date on record, make no changes to the `SocialNetwork` and set the status to `INVALID_DATE`.

These methods:

- Throw a `NullPointerException` if any argument is null.
- In case of success, set the status to `SUCCESS`.

Additionally, these classes may contain as many auxiliary private methods as you see fit, and additional helper classes may be defined.

Discussion Guidelines

- High-level code organization
- The design and documentation of the social network implementation
- Functions that exceed McCabe's complexity of 4 (if any)
- Non-structured programming constructs and break statements (if any)

On-Line Submission

Turn in with your code:

- A README file explaining how to compile and run the code.
- A Make or Ant build file.
- A comment at the top over file containing your name, email address, a one-sentence description of the file, and if necessary a longer comment describing the design of the file.

The code should be handed in a zip, tar.bz2, or tar.gz archive. Archives in 7z cannot be accepted.