Programming Assignment 1

CS200 Spring 2012

1

NeesFeed

Every item that shows up in the News Feed page is an Object

Whenever another user interacts with an object, an EDGE will be created between the user and that object.

How to build New Feeds ?

- EdgeRank: Algorithm to build a list of news items that are newer and more relevant.

3

Optimizing the News Feed

livestream

6. NFO: News Feed Optimization

EdgeRank

$$\sum_{edges\ e} u_e w_e d_e$$

u_e - affinity score between viewing user and edge creator

w_e - weight for this edge type (create, comment, like, tag, etc.)

d_e - time decay factor based on how long ago the edge was created

4

Let's Break Down the Task

Assignment #	Objectives	Features
PA 1 (Individual)	Single-person SN	1. Personal info 2. Managing Edges
PA2 (Team)	Calculate ER value and sort them	1. Calculating ER score 2. Sorting (Quicksort)
PA3 (Team)	Extend to multi-user SN	1. Queue of Friends 2. Binary tree of members
PA4 (Team)	Managing multiple members	1. Build information
PA5 (Team)	Implementing the News Feed list	1. Calculate ER score for specific person's friend's edge 2. List of news feed

5

Programming Assignment 1

- Building a *Single-person-Social Network Application*
- Due date: **Feb,2,2:00PM**
- Late submission: Feb,3, 2:00PM
  - 10% Score deduction
    - No submission allowed after Feb. 3, 2:00 pm
- **Individual** submission

6

Objective

- Build classes to manage information for a single user.
  - Users name (first name and last name)
  - userID
  - Only the user post's on their wall
    - For this assignment.
- There is no "*friend*" yet.
  - Later assignments will expand to let the user's friends post on the wall.

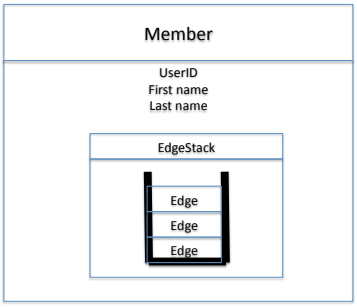
7

Tasks

- You will be creating 5 classes
  - `Member`
  - `EdgeStack`
  - `Edge`
  - `Item`
  - `InformationParser`
- `Modify 1 class`
  - `PA1`

8

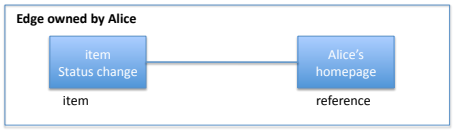
Managing a member



9

Edges and NewsFeed Objects

- Alice posted a new photo.
  - `Item` change-status is created by Alice
    - Alice's wall (homepage) is reference
    - Change-status is item



10

Edges and **NewsFeed** Objects

- Bob commented on Alice's change-status
  - `For this Edge`
    - Bob's comment is the item
    - Alice's change-status is the reference

11

Member

- Provides methods to retrieve information and update information
  - userID: unique ID for this user
  - First name: user's first name
  - Last name: user's last name
  - edgestack: a class maintaining a stack of edges

12

Member

```
public class Member {
     private String userID, first, last;
     private EdgeStack edgeStack;
     public Member(){}
     public Member(String _userID, String _first, String _last){}
     public String getUserID(){}
     public String getFirst(){}
     public String getLast(){}
     public EdgeStack getEdgeStack(){}
     public void setUserID(String _userID){}
     public void setFirst(String _first){}
     public void setLast(String _last){}
     public void setEdgeStack(EdgeStack _edgeStack){}
}
```

13

EdgeStack

- Stack of Edges
  - Class that maintains a data structure to store and retrieve edges
    - isEmpty()
    - push()
    - pop()
    - peek()
    - create()
    - destroy()

14

Implementing with java.util.LinkedList

```
public class EdgeStack {
  private LinkedList<Edge> llist;
  public EdgeStack(){}
  public boolean isEmpty(){}
  public void push(Edge e){}
  public Edge pop(){}
  public Edge peek(){}
  public LinkedList popAll(){}
}
```

15

Edge: Maintains information about an edge

- `item`: the object *created* as a result of this activity
  - e.g. the comment message added
- `reference`: the object *related* to this activity
  - e.g. the wall posting, status-change

16

Edge

```
public class Edge {
  private Item item;
  private Item reference;
  public Item getItem(){}
  public Item getReference(){}
  public void setItem(Item item){}
  public void setReference(Item item){}
.
.
.
}
```

17

Item

- Class maintains information about any object shown on the NewsFeed page. (e.g. posting, comment, picture, etc.)
- `itemID`: unique id for this object
- `type`: type of the newsfeed object
  1: wall posting 2: comment 3: homepage 4: status change 5: like 6: visit
- `creator`: member ID of the original author
- `message`: the content
- `age`: age of the item (in second)

18

NFObject

```
public class Item {
   private int itemID;
   private int type;
   private String creator;
   private String message;
   private int age;
   public Item (int itemId, int type, String creator, String
   message, int age){};
   public int getItemID(){};
   public int getType(){};
   public String getCreator(){};
   public String getMessage(){};
   public int getAge(){};
  …..
}
```

19

InformationParser

- We use synthesized data to simplify it.
- Data file is provided: PA1-exampleData.txt
- `InformationParser` should ***read/parse/ create*** object from this example file.

20

Format of Example File

```
member [member id] [first name] [last name]
edge
item [itemID][type][creator][msg][age]
reference[itemID][type][creator][msg][age]
edge
item [itemID][type][creator][msg][age]
reference[itemID][type][creator][msg][age]
edge
item [itemID][type][creator][msg][age]
reference[itemID][type][creator][msg][age]
```

21

Parsing Process

- Step 1. **Open** the file
- Step 2. **Read** a line
- Step 3. **Determine the types** of information
- Step 4. **Break the line** into information
- Step 5. **Store** the information in an object
- Step 6. **Return** the object

22

Useful java classes for parsing

- `java.io.File`
- `java.util.Scanner`
- `java.lang.String`
  - `split()` : **tokenize** the string
  - `equals()`: detect **keywords**

23

Example

```
String[] result = "this is a test".split("\
  \s");
for (int x=0; x<result.length; x++)
  System.out.println("result["+x+"] "+result
  [x]);
```

```
result[0] this
result[1] is
result[2] a
resutl[3] test
```

24

PA1.java

- This is the class that includes the `main()` for this assignment.
- **Do NOT modify the `main()` method**. We will use this directly for grading.

25

`Main()` in PA1.java

```
public static void main(String args[]){
     String input_file = args[0];
     String cmd = args[1];
     PA1 pa1 = new PA1(input_file);

     //test case 1: print name
     if (cmd.equals("print_name")){
               pa1.print_name();
     } else if (cmd.equals("print_userID")){
               //test case 2: print user ID
               pa1.print_userID();
     }else if(cmd.equals("pop_print_EdgeStack")){
               //test case 3: retrieve the nth recently added edge
               //and print current stack
               pa1.pop_print_EdgeStack();
     }else if (cmd.equals("peek_print_Edge")){
               //test case 4:pick nth recently added edge
               //and print that edge
               pa1.peek_print_Edge ();
     }
     }else if (cmd.equals("peek_print_EdgeStack")){
               //test case 5: pick nth recently added edge
               //and print current stack
               pa1.peek_print_EdgeStack ();

}
```

26

How to design your software?

- Option 1: Use of the provided skeleton files is optional.
- Option 2: You are encouraged to add more data structure or design patterns (e.g. inheritance)
- Follow the test case scenario. (Do not modify main() in PA1.java)
- Search examples, read them, and try to apply them in your software.
- Ask your TA!

27

Submitting your PAs

- Use checkin
- Make **a tar ball** with your source code.
  - TA will be looking at your code and design.
- It should compile with following command in the directory you stored your source code
  - e.g. `javac *.java`
  - Make sure that your source code files are in one flat directory.
- If you did not work on the CS cluster, please make sure that your code works in the **CS linux cluster**.
  - You can ssh to CS machines.

28