Lösningsförslag, kontrollskrivning 3 PTDC

2013–12–16

1.
```
public class URand {
    private int[] nbrs;
    private int current;
    private Random rand;

    public URand(int n) {
        nbrs = new int[n];
        for (int i = 0; i < n; i++) {
            nbrs[i] = i;
        }
        rand = new Random();
        shuffle();
    }

    public int nextInt() {
        if (current == nbrs.length) {
            shuffle();
        }
        int res = nbrs[current];
        current++;
        return res;
    }

    private void shuffle() {
        for (int i = nbrs.length - 1; i >= 1; i--) {
            int j = rand.nextInt(i + 1);
            int tmp = nbrs[i];
            nbrs[i] = nbrs[j];
            nbrs[j] = tmp;
        }
        current = 0;
    }
}
```

2.
```
public class Accumulator {
    private ArrayList<Integer> stack;
    private int sum;

    public Accumulator() {
        stack = new ArrayList<Integer>();
        sum = 0;
    }

    public int getSum() {
        return sum;
    }
    public void add(int nbr) {
        sum += nbr;
        stack.add(nbr);
    }

    public void undo() {
        if (!stack.isEmpty()) {
            int top = stack.size() - 1;
            sum -= stack.get(top);
            stack.remove(top);
        }
    }

    public void commit() {
        stack.clear();
    }

    public void rollback() {
        while (!stack.isEmpty()) {
            undo();
        }
    }
}
```

3.
```
public class Entry {
    private String key;
    private String value;

    public Entry(String key, String value) {
        this.key = key;
        this.value = value;
    }

    public String getKey() {
        return key;
    }

    public String getValue() {
        return value;
    }
}

public class Properties {
    private ArrayList<Entry> props;
    private Properties defaults;

    public Properties(Properties defaults) {
        this.defaults = defaults;
        props = new ArrayList<Entry>();
    }

    public Properties() {
        this(null);
    }
    public String getProperty(String key) {
        String result = null;
        int pos = find(key);
        if (pos >= 0 ) {
            result = props.get(pos).getValue();
        } else if (defaults != null) {
            result = defaults.getProperty(key);
        }
        return result;
    }

    public void setProperty(String key, String value) {
        int pos = find(key);
        if (pos >= 0) {
            props.set(pos, new Entry(key, value));
        } else {
            props.add(new Entry(key, value));
        }
    }

    private int find(String key) {
        for (int i = 0; i < props.size(); i++) {
            if (props.get(i).getKey().equals(key)) {
                return i;
            }
        }
        return -1;
    }

    public void load(Scanner scan) {
        while (scan.hasNext()) {
            String line = scan.nextLine();
            int eqPos = line.indexOf('=');
            String key = line.substring(0, eqPos);
            String value = line.substring(eqPos + 1, line.length());
            props.add(new Entry(key, value));
        }
    }

    public void store(PrintWriter file) {
        for (int i = 0; i < props.size(); i++) {
            Entry e = props.get(i);
            file.println(e.getKey() + "=" + e.getValue());
        }
        printer.close();
    }
}
```