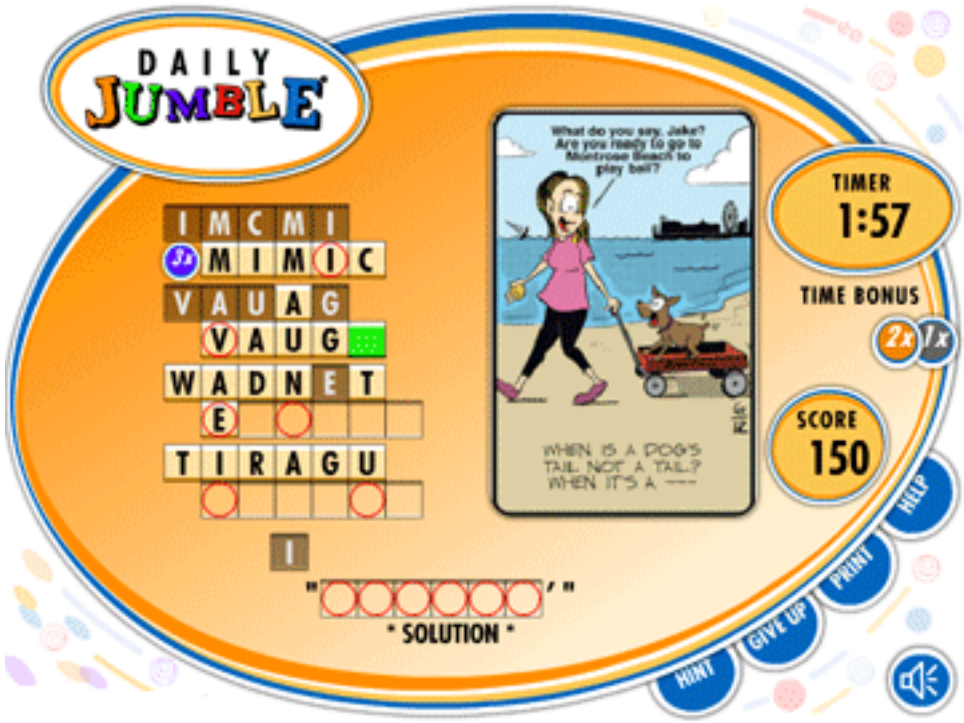
Assignment 2
Due Feb 11

1. jumble.java



The *JUMBLE* is a puzzle that rearranges the letters of a word. Normally your task is to unscramble the letters and discover the original word. However, for this assignment you will take a word and create a jumble or mixed up version.

Write a program that accepts a word and produces a "jumbled" version. For example, the letters of the word "TAKEN"
might be scrambled as "AKNET".

You can scramble the letters of a word by repeatedly exchanging a randomly a chosen letter with the first letter. Ten exchanges is more than enough to jumble a 5 or 6 letter word. **Use the Random class**

Use StringBuilder and the method setCharAt(int i)
Use a static method (as you did last semester)to do the work and return the jumbled word.

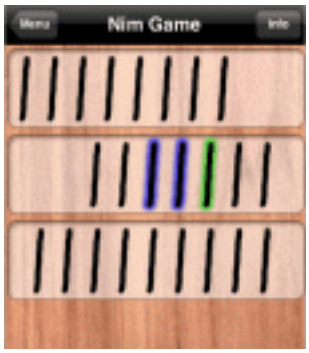
Do not do the jumbling in main(...);

Use this as your main method:

```
public static void main(String[] args)
{
    Scanner input = new Scanner(System.in);
    System.out.print("Enter a word use XXX to end the program: ");
    String word = input.nextLine();
    while (!word.equals("XXX")
    {
        System.out.println("The jumbled word is "+jumble(word));
        System.out.print("Enter a word use XXX to end the program: ");
        word = input.nextLine();
     }
}
```

2. This program uses classes as we did on Thursday. Use constructors and the methods are not static.

Write an application that allows two players to play the game of Nim. The game board for Nim consists of any number of piles of sticks, (or pencils, matches, stones, or coins). And, each pile contains an arbitrary number of sticks. Players take turns removing sticks from a single pile. A player can remove any number of sticks at his/her turn, but only from one pile. The player to remove the last stick wins the game.

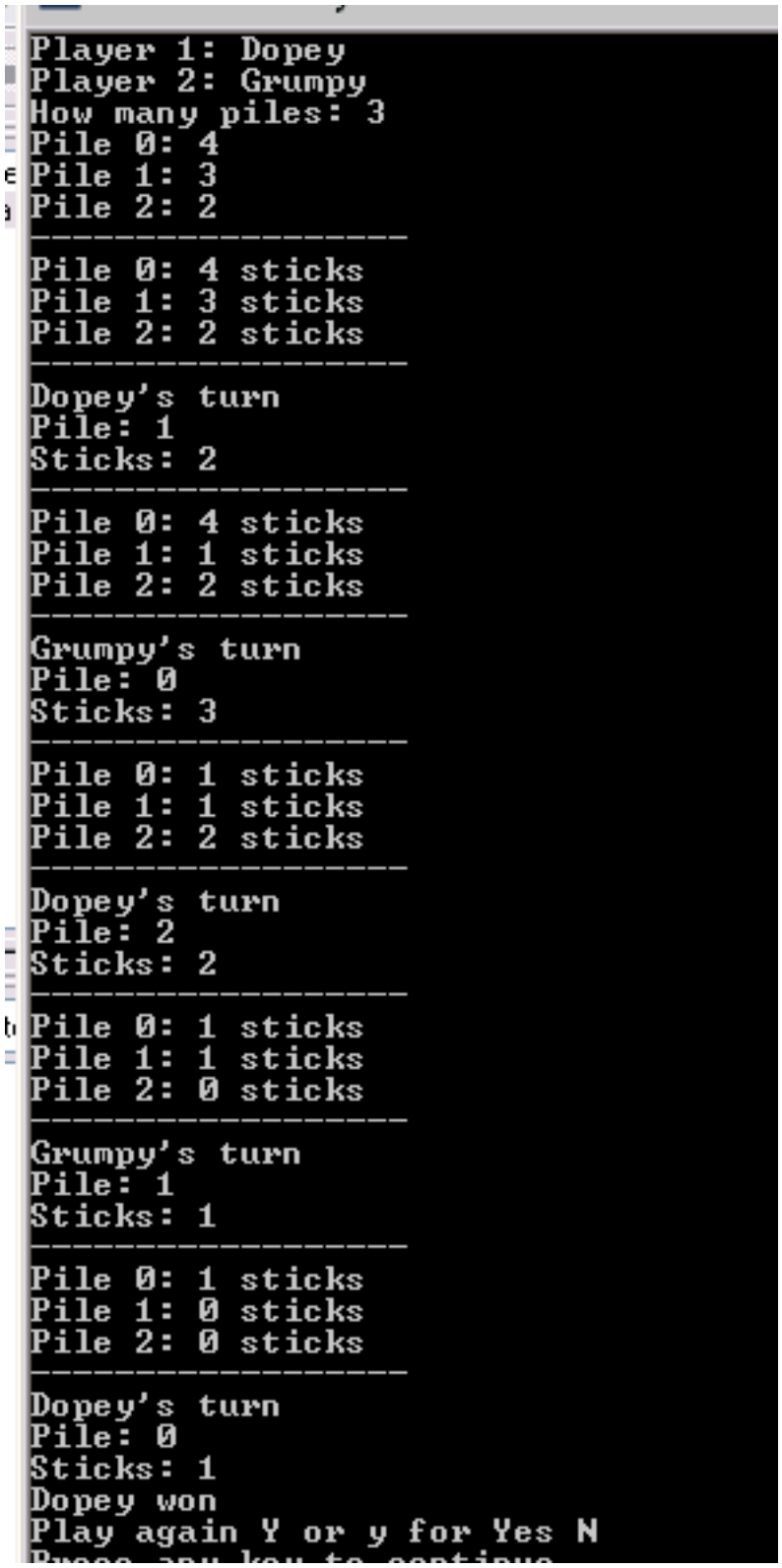


The application should ask each player to enter his/her name and then play the game.
The first player begins the game.

When a game is over, the application should display a message stating which player won, and ask the players if they would like to play again.

Note: Allow only legal moves. That is, a player must remove at least one stick and may not remove more sticks than are presently in a pile.

Here is how playing the game should look:



To implement the application use **two classes**:

Game and PlayGame

The Game class models the gameboard.

The Game class should have two private variables

```
int num Piles;    // number of piles
int [] piles      // an array holding the number of sticks in each pile
```

A default constructor should initialize

numPiles to 3 and put 10 sticks in each pile.

A two-argument constructor should have parameters that supply the number of piles and an array with the number of sticks in each pile:

```
public Game( int num, int[] sticks)
{

}
```

There should be public methods

public void remove(int pile, int numSticks) // pile is pile number, numSticks is how many to remove

public boolean gameOver()

public void printPiles() // prints the pile number and the number of sticks currently in a pile.

The PlayGame class has private data

```
String player1, player2;      // names of the players
int turn;                     // whose turn player 1 or player 2, initially 1
```

Game game; // a Game object (actually a reference to a Game object)
Scanner input;

The only constructor is the default constructor, which should

- ask each player his/her name
- set turn to 1
- Ask how many piles in the game (num piles)
- Ask how many stick per pile and build an array int piles[]
- Instantiate a new Game object with the number of piles and the array of sticks

  game = new Game(numPiles, piles)

You should also have a method

public void play() that plays the game until no sticks remain. Remember to switch turn back and forth between 1 and 2.

The main method of PlayGame is very simple

1. instantiate a PlayGame object:

   PlayGame game = new PlayGame()

2. play the game :

   game.play()

Include a loop that asks if the player wants to play again or not.

These classes should be in two files: Game.java and PlayGame.java