Honor Code
Like all work for this class, the Emory Honor Code applies. You should do your own work on all problems, unless you are explicitly instructed otherwise. If you get stuck or have questions, ask your instructor or a TA for help. For EACH of the files you submit, be sure to put the appropriate honor code statement (as specified on the course syllabus) at the top of the file in comments. Otherwise, you will be assessed a 10 point deduction.

Preparation
1. Create a directory called hw6 inside your cs170 project directory to save your hw6 files.
   
   mkdir ~/cs170/hw6

2. You must use /cs170/hw5 directory as your current directory when editing any program files for hw5. Copy the files from the shared directory:
   
   cp /home/cs170002/share/hw6/* ~/cs170/hw6/

   Change your current directory to your newly created hw6 directory:
   
   cd ~/cs170/hw6/

3. You can now run gedit to edit your program:
   
   gedit SnakesAndLadders.java &

Problem
SnakesAndLadders
The objective of this program is practice the use of methods and arrays by simulating the classic game of Snakes and Ladders. The program asks the user for the number of players, and then the players will automatically roll the die and move forward in the board. If a player falls in a square with a lower end of a ladder, then the player moves to the upper end. If a player falls in a square with the head of snake, then the player goes down to the square with the tail of the snake. The following is an image of the board:
The game ends when a player gets to square 100 or above. The following could be a sample run of the program:

How many players?
2
Player 0 just rolled: 1 and now is in position: 1
Player 1 just rolled: 4 and now is in position: 4
This is a ladder! New position is: 14
Player 0 just rolled: 2 and now is in position: 3
Player 1 just rolled: 3 and now is in position: 17
This is a Snake! New position is: 7
Player 0 just rolled: 1 and now is in position: 4
This is a ladder! New position is: 14
Player 1 just rolled: 2 and now is in position: 9
This is a ladder! New position is: 31
Player 0 just rolled: 1 and now is in position: 15
Player 1 just rolled: 1 and now is in position: 32
Player 0 just rolled: 5 and now is in position: 20
This is a ladder! New position is: 38
Player 1 just rolled: 5 and now is in position: 37
Player 0 just rolled: 2 and now is in position: 40
This is a ladder! New position is: 59
...
Player 1 just rolled: 5 and now is in position: 100
Game over! Player 1 wins!!
Problem Decomposition

Create a boolean method `hasAnybodyWon` that, given an array of players, returns true if anybody has won (i.e. is in square 100 or above) or false otherwise. Run your program while this is false (or, in other words, nobody has won). Create an int method that simulates the rolling of a die, so it returns a random, integer number between 1 and 6. Also, create an int method `giveIndex`, that, given an array and a number, returns the position in which the number is in the array. If the number is not in the array, return -1. A couple of these methods have already been initialized for you. At each turn, make the players roll the die and move the appropriate amount of squares. Check if they have fallen in a square with either the bottom of a ladder or the head of a snake, and move the player accordingly. Move on to the next player and repeat the procedure while nobody has won yet. Print appropriate messages when needed. Extra (+5). If a player rolls a 6, this player rolls again. However, if she rolls three 6's in a row, the player goes back to the first square and it is the next player's turn. A few hints are provided in the code. Here is a sample run if you incorporate this feature:

How many players?
2
Player 0 just rolled: 4 and now is in position: 4
This is a ladder! New position is: 14
Player 1 just rolled: 6 and now is in position: 6
Player 1 just rolled: 6 and now is in position: 12
Player 1 just rolled: 6 and now is in position: 18
Oh no! three 6’s in a row! Player 1 goes back to the first square!!
Player 0 just rolled: 2 and now is in position: 16
Player 1 just rolled: 4 and now is in position: 4
This is a ladder! New position is: 14
Player 0 just rolled: 6 and now is in position: 22
Player 0 just rolled: 3 and now is in position: 25
Player 1 just rolled: 6 and now is in position: 20
Player 1 just rolled: 6 and now is in position: 26
Player 1 just rolled: 5 and now is in position: 31
Player 0 just rolled: 3 and now is in position: 28
...
Player 0 just rolled: 5 and now is in position: 102
Game over! Player 0 wins!!
Grading

Your grade will be determined based on the correctness of your programs as well as program style. Program style includes such things as comments, variable/method names, and readability. Late penalties apply as per the class syllabus.

Submission

- When you finish, you should have one file in your cs170/hw6 directory: SnakesAndLadders.java.

- Be sure to include the Honor Code statement at the top of EACH file you submit.

- Only your last submission will be graded.

- Submit your work using the following commands. You need to be in your /cs170/hw6 directory when you issue them:

  ```bash
  $/home/cs170002/turnin-hw SnakesAndLadders.java hw6
  ```

- Your homework is not turned unless the above commands are successful (you will get a "success" message when turn in was successful).