import java.util.HashMap;  
import java.util.Random;

public class SnakenChutes {  
   int numofPlayers;  
   int[] players;  
   HashMap<Integer,Integer> ladder;  
   HashMap<Integer,Integer> chutes;  
    
   SnakenChutes(int numofPlayers){  
       this.numofPlayers=numofPlayers;  
       players=new int[numofPlayers]; //Creating an array to keep track of players advances  
       ladder=new HashMap<Integer,Integer>(); //HashMap for ladder to keep start and end linked  
       setupLadder(ladder); //calling the setup to add entries to ladder HashMap  
       chutes=new HashMap<Integer,Integer>(); //HashMap for chutes to keep start and end linked  
       setupChutes(chutes);//calling the setup to add entries to chutes HashMap  
   }  
    
   public void setupLadder(HashMap<Integer,Integer> temp){  
       temp.put(1,38);  
       temp.put(4,14);  
       temp.put(9,31);  
       temp.put(28,84);  
       temp.put(21,42);  
       temp.put(36,44);  
       temp.put(51,68);  
       temp.put(71,91);  
       temp.put(80,100);  
        
   }  
    
   public void setupChutes(HashMap<Integer,Integer> temp){  
       temp.put(98,78);  
       temp.put(95,75);  
       temp.put(93,73);  
       temp.put(87,24);  
       temp.put(64,60);  
       temp.put(62,19);  
       temp.put(56,53);  
       temp.put(49,11);  
   }  
    
   public int diceRoll(){  
       Random r=new Random(); //Rolling the dice  
       return (r.nextInt(6))+1;  
        
   }  
    
   public int isLadder(int pos){ //Checking if the present position is starting pt. for ladder  
       if(ladder.containsKey(pos)){  
           System.out.println("Ladder used for below");  
           return ladder.get(pos);  
       }  
        
       return   isChutes(pos); //calling the isChutes if the present position is not ladder found  
        
            
    
   }  
    
   public int isChutes(int pos){ //Checking if the present position is starting pt. for chutes  
       if(chutes.containsKey(pos)){  
           System.out.println("Chutes used for below");  
           return chutes.get(pos);  
       }  
       return pos;  
    
   }  
    
    
    
   public boolean isWinner(int val){ //Checking for any winners  
       if(val==100)  
           return true;  
           else  
               return false;  
   }  
    
   public boolean checkMax(int val){  
       if(val > 100){  
           return true;  
            
       }  
       else  
           return false;  
   }  
    
   public void play(){  
        
       int playerChance=0;  
       while(true){  
           playerChance=playerChance%numofPlayers; //Keep of rotating between players till a winner is found  
           int roll=diceRoll();  
           int prev=players[playerChance]; //GEtting previous position  
           int val=isLadder(prev+roll); //New position  
           int temp=0;  
           if(isWinner(val)){  
               System.out.println("Player "+(playerChance+1)+" is winner");  
               return;  
                
           }  
           else{  
               if(checkMax(val)){  
                   temp=prev;  
                   players[playerChance]=prev;  
                   playerChance++;  
               }  
               else{  
                   temp=val;  
                   players[playerChance]=val; //Saving new position  
                   playerChance++;  
               }  
                
           }  
           System.out.println("Player "+(playerChance+1)+" chance, was at "+prev+" now got a no."+roll+ " now at "+temp);  
            
       }  
   }  
}

Below is the code for SnakeMain.java

public class SnakeMain {  
    
   public static void main(String[] args){  
       SnakenChutes c=new SnakenChutes(4);  
       c.play();  
   }

}