ASDE Algorithm Test

Imagine Abhimanyu in Chakravyuha. There are 11 circles in the Chakravyuha surrounded by different enemies. Abhimanyu is located in the innermost circle and has to cross all the 11 circles to reach Pandavas army back.

Given:

* Each circle is guarded by different enemy where enemy is equipped with k1, k2……k11 powers Abhmanyu start from the innermost circle with p power Abhimanyu has a boon to skip fighting enemy a times
* Abhmanyu can recharge himself with power b times
* Battling in each circle will result in loss of the same power from Abhimanyu as the enemy.
* If Abhmanyu enter the circle with energy less than the respective enemy, he will lose the battle
* k3 and k7 enemies are endured with power to regenerate themselves once with half of their initial power and can attack Abhimanyu from behind if he is battling in iteratively next circle

Write an algorithm to find if Abhimanyu can cross the Chakravyuh and test it with two sets of test cases.

**Algorithm**

**Input Collection:**  
Gather the energies of those enemies in every one of the 11 circles in the game.  
Collect Abhimanyu's initial power.  
Gather the possible number of times shown by Abhimanyu that he is able to skip a battle.  
Tally the amount by which Abhimanyu can charge the power up.

**Initialize:**  
Create an array regenerated to keep track of whether enemies in circles 3 and 7 have regenerated.

**Iterate Through Circles:**  
  
For each circle from 0 to 10 (representing circles 1 to 11):  
Set the logic check if the current circle is circle 3 or circle 7.  
The algorithm must check whether the enemy in these circles has not regenerated yet .The enemy’s power must be halved to regenerate them, and the form of the regenerated array must be updated.  
If Abhimanyu's power is greater than the enemy's power, he fights and reduces his power by the enemy's power.  
If Abhimanyu's power is not enough but he can skip the battle, use a skip.

If he cannot skip but can recharge, use a recharge.  
If none of the above conditions are met, Abhimanyu loses the battle, print a message, and return false.

Print Abhimanyu's remaining power after each circle

**Final Check:**  
  
If Abhimanyu is able to successfully complete the task of crossing all circles, then print a success message and return true.

**Testing**

**Test Case 1**:

Enemy powers: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110

Initial power of Abhimanyu: 200

Number of skips: 2

Number of recharges: 1

**Output**

After circle 1, Abhimanyu's power is 190

After circle 2, Abhimanyu's power is 170

After circle 3, Abhimanyu's power is 140

After circle 3, Abgimanyu's power after defeating the regenerated enemy is 125

After circle 4, Abhimanyu's power is 85

After circle 5, Abhimanyu's power is 35

After circle 6, Abhimanyu's power is 35

Abhimanyu lost the battle at circle 7 due to attack from behind.

**Test Case 2:**

Enemy powers: 15, 25, 35, 45, 55, 65, 75, 85, 95, 105, 115

Initial power of Abhimanyu: 250

Number of skips: 1

Number of recharges: 2

**Output**

After circle 1, Abhimanyu's power is 235

After circle 2, Abhimanyu's power is 210

After circle 3, Abhimanyu's power is 175

After circle 3, Abgimanyu's power after defeating the regenerated enemy is 158

After circle 4, Abhimanyu's power is 113

After circle 5, Abhimanyu's power is 58

After circle 6, Abhimanyu's power is 58

Abhimanyu lost the battle at circle 7 due to attack from behind.

**Code snippet**

import java.util.Scanner;

public class Abhimanyu {

public static boolean chakravyuhaBattle(int[] k, int p, int a, int b) {

int n = k.length;

boolean[] regenerated = new boolean[n];

for (int i = 0; i < n; i++) {

if (i == 2 || i == 6) {

if (!regenerated[i]) {

regenerated[i] = true;

k[i] = k[i];

if (i > 0 && p <= k[i]) {

System.*out*.println("Abhimanyu lost the battle at circle " + (i + 1) + " due to attack from behind.");

return false;

}

}

}

if (p > k[i]) {

p -= k[i];

} else if (a > 0) {

a--;

} else if (b > 0) {

p += k[i];

b--;

} else {

System.*out*.println("Abhimanyu lost the battle at circle " + (i + 1));

return false;

}

System.*out*.println("After circle " + (i + 1) + ", Abhimanyu's power is " + p);

if ((i == 2 && regenerated[i]) || (i == 6 && regenerated[i])) {

k[i] = k[i]/2;

if (p <= k[i]) {

System.*out*.println("Abhimanyu lost the battle at circle " + (i + 1) + " due to attack from behind.");

return false;

}

else {

p-=k[i];

System.*out*.println("After circle "+ (i+1)+ ", Abgimanyu's power after defeating the regenerated enemy is " + p);

}

}

}

System.*out*.println("Abhimanyu successfully crossed all circles and reached the Pandavas army.");

return true;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.*in*);

int[] k = new int[11];

System.*out*.println("Enter the power of enemies in each of the 11 circles:");

for (int i = 0; i < 11; i++) {

k[i] = scanner.nextInt();

}

System.*out*.println("Enter the initial power of Abhimanyu:");

int p = scanner.nextInt();

System.*out*.println("Enter the number of times Abhimanyu can skip a battle:");

int a = scanner.nextInt();

System.*out*.println("Enter the number of times Abhimanyu can recharge:");

int b = scanner.nextInt();

*chakravyuhaBattle*(k, p, a, b);

scanner.close();

}

}