**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.

-------------------------------------------------------------------------------------------------------------------------------

**Assumptions**(as they were not clear in the problem statement):

- Abhimanyu can kill an enemy that has the same amount of energy as he currently has.

- Recharging means going up to the initial energy level

**Solutions:**

bool canAbhimanyuCrossChakravyuhaRecursive(int i, int a, int b, int power, int p, vector<int> &k) {

if (i >= (int)k.size()) {

return true;

} else {

bool possible = false;

if (a > 0) {

possible |= canAbhimanyuCrossChakravyuhaRecursive(i + 1, a - 1, b, power, p, k);

}

if (power < k[i]) {

if (p >= k[i]) {

power = p;

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

k[i + 1] += k[i] / 2;

}

possible = possible || canAbhimanyuCrossChakravyuhaRecursive(i + 1, a, b - 1, power - k[i], p, k);

}

} else {

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

k[i + 1] += k[i] / 2;

}

possible = possible || canAbhimanyuCrossChakravyuhaRecursive(i + 1, a, b, power - k[i], p, k);

}

return possible;

}

}

**Test cases 1:**

a = 4, b = 5, p = 3, k[]={2,8,5,12,3,2,7,10,4,6,11}

Output : 0

**Test cases 2:**

a = 7, b = 6, p = 4, k[]={6,1,4,12,3,7,2,9,5,8,1}

Output **:** 1