**Problem statement: Can Abhimanyu cross the Chakravyuh or not !!**

**Given:**

* Abhimanyu starts with **p** power.
* There are 11 circles with enemy power **K1, K2, K3, …, K11**.
* Abhimanyu can skip fighting **a** times.
* Abhimanyu can recharge and gain power **b** times.

**More Information about battling:**

* While moving through each circle, if Abhimanyu’s power **p**  >= Enemies power Ki, he can defeat the enemy. His power is decreases by Ki.
* If Abhimanyu’s power is less than **Ki,** he losses the battle unless he skips that enemy using one of his skips (out of **a** times).
* Enemies at positions **K3** and **K7** regenerate once. After the first defeat, they regenerate with half of their initial power. They can also attack from behind while Abhimanyu is battling in next circle.
* Abhimanyu can recharge his power **b** times. Since in problem statement it’s not mentioned, so I am assuming recharging restores full power **p** each time.

**Algortihm:**

* Iterate through all 11 circles:

– If Abhimanyu’s power is greater than or equal to enemy’s power i.e. **Ki**, he defeats the enemy and loses his power equal to **Ki.**

**–** If Abhimanyu’s power is less than the enemy’s, he has two choices:

– Either skip the enemy, if skips are available or Recharge his power, if recharges  
 are available.

– Loss.

* For **K3** and **K7** enemies:

– After Abhimanyu defeats them the first time, they regenerate with half of their initial power.

– They can attack from behind while Abhimanyu is in the next circle, so have to consider these attacks too.

* At the end if Abhimanyu crosses all the circle with power greater than zero, wins otherwise Loss.

**Pseudocode:**

* Let **p** be the Abhimanyu’s initial power, **enemies** be the list of strength of each enemy of each circle, **a** be the skips available and **b** be the recharges remaining.

function **canAbhimanyuCrossChakravyuh**(p, enemies, a, b){

skipsAvailable = a;

rechargesRemain = b;

currentPower = p;

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

if(i == 2 or i == 6){ // for the case of K3 and K7

if(currentPower >= enemies[i]){

currentPower -= enemies[i];

regeneratedPower = enemies[i]/2; // for the case of regeneration.

if(currentPower < regeneratedPower){

//attacks from behind.

if(skipsAvailable > 0){

skipsAvailable -= 1;

}else if (rechargesRemain > 0){

rechargesRemain -= 1;

currentPower = p; //recharging power.

}else{

return false;

}

}

}else{

if(skipsAvailable > 0){

skipsAvailable -= 1;

}else if(rechargesRemain > 0){

rechargesRemain -= 1;

currentPower = p;

}else{

return false;

}

}

}else{

if(currentPower >= enemies[i]){

currentPower -= enemies[i];

}else{

if(skipsAvailable > 0){

skipsAvailable -= 1;

}else if(rechargesRemain > 0){

rechargesRemain -= 1;

currentPower = p;

}else{

return false;

}

}

}

}

return true;

}

**Code in C++:**

#include <iostream>

#include <vector>

using namespace std;

bool canAbhimanyuCrossChakravyuh(int p, vector<int>& enemies, int a, int b) {

int skipsAvailable = a; // Skips available

int rechargesRemain = b; // Recharges available

int currentPower = p; // Abhimanyu's current power

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

if (i == 2 || i == 6) { // Special case for k3 and k7

if (currentPower >= enemies[i]) {

currentPower -= enemies[i]; // Battle and lose power

int regeneratedPower = enemies[i] / 2; // Regenerated power

if (currentPower < regeneratedPower) {

// Attack from behind

if (skipsAvailable > 0) {

skipsAvailable--; // Use a skip

} else if (rechargesRemain > 0) {

rechargesRemain--; // Use a recharge to restore full power

currentPower = p;

} else {

return false; // Abhimanyu loses

}

}

} else {

// Not enough power to defeat k3 or k7 enemy

if (skipsAvailable > 0) {

skipsAvailable--; // Use a skip

} else if (rechargesRemain > 0) {

rechargesRemain--; // Use a recharge to restore full power

currentPower = p;

} else {

return false; // Abhimanyu loses

}

}

} else {

// For all other enemies

if (currentPower >= enemies[i]) {

currentPower -= enemies[i]; // Battle and lose power

} else {

// Not enough power to defeat the enemy

if (skipsAvailable > 0) {

skipsAvailable--; // Use a skip

} else if (rechargesRemain > 0) {

rechargesRemain--; // Use a recharge to restore full power

currentPower = p;

} else {

return false; // Abhimanyu loses

}

}

}

}

return true; // Abhimanyu successfully crosses all circles

}

int main() {

// Test case 1

int p1 = 10;

vector<int> enemies1 = {0, 10, 2, 3, 9, 5, 6, 7, 8, 4, 1};

int a1 = 6;

int b1 = 2;

if (canAbhimanyuCrossChakravyuh(p1, enemies1, a1, b1)) {

cout << "Abhimanyu can cross the Chakravyuh.\n";

} else {

cout << "Abhimanyu cannot cross the Chakravyuh.\n";

}

// Test case 2

int p2 = 50;

vector<int> enemies2 = {15, 25, 35, 45, 55, 65, 75, 85, 95, 105, 115};

int a2 = 1;

int b2 = 1;

if (canAbhimanyuCrossChakravyuh(p2, enemies2, a2, b2)) {

cout << "Abhimanyu can cross the Chakravyuh.\n";

} else {

cout << "Abhimanyu cannot cross the Chakravyuh.\n";

}

return 0;

}