import java.util.\*;

public class MyClass {

public static int bfs(ArrayList<Integer>[] country, HashSet<Integer> hotspots, int city, int[] distance) {

int farestHotspot = 0, connetedCity;

Queue<Integer> queue = new LinkedList<Integer>();

queue.add(city);

distance[city] = 0;

while (!queue.isEmpty()) {

city = queue.poll();

if (hotspots.contains(city))

farestHotspot = city;

for (int i = 0; i < country[city].size(); i++) {

connetedCity = country[city].get(i);

if (distance[connetedCity] == -1) {

distance[connetedCity] = distance[city] + 1;

queue.add(connetedCity);

}

}

}

return farestHotspot;

}

public static int numberOfEpicentres(ArrayList<Integer>[] country, int n, HashSet<Integer> hotspots, int h, int x) {

int farHotspot1, farHotspot2, i, count = 0;;

int[] dist\_RandomCity = new int[n], dist\_FarHotspot1 = new int[n], dist\_FarHotspot2 = new int[n];

Arrays.fill(dist\_RandomCity, -1);

Arrays.fill(dist\_FarHotspot1, -1);

Arrays.fill(dist\_FarHotspot2, -1);

farHotspot1 = bfs(country, hotspots, 0, dist\_RandomCity);

farHotspot2 = bfs(country, hotspots, farHotspot1, dist\_FarHotspot1);

bfs(country, hotspots, farHotspot2, dist\_FarHotspot2);

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

if (dist\_FarHotspot1[i] <= x && dist\_FarHotspot2[i] <= x)

count++;

}

return count;

}

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

int i, city1, city2, n = sc.nextInt(), h = sc.nextInt(), x = sc.nextInt();

HashSet<Integer> hotspots = new HashSet<>();

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

hotspots.add(sc.nextInt()-1);

}

int roads[][] = new int[n-1][2];

for (i = 0; i < n-1; i++){

roads[i][0] = sc.nextInt();

roads[i][1] = sc.nextInt();

}

ArrayList<Integer>[] country = new ArrayList[n];

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

country[i]= new ArrayList<Integer>();

}

for (i = 0; i < (n - 1); i++) {

city1 = roads[i][0] - 1;

city2 = roads[i][1] - 1;

country[city1].add(city2);

country[city2].add(city1);

}

System.out.println(numberOfEpicentres(country, n, hotspots, hotspots.size(), x));

}

}