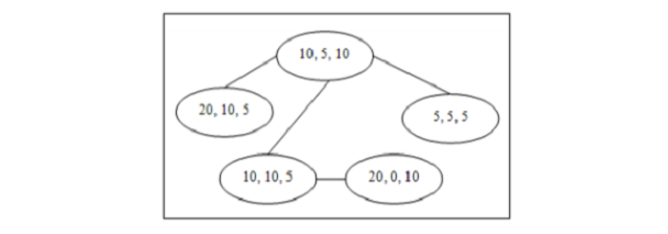
Assignment 5

2. Castle

Wars have played a significant role in world history. Unlike modern wars, armies in the middle ages were principally concerned with capturing and holding castles, the private fortified residences of lords and nobles. The size of the attacking army was an important factor in an army’s ability to capture and hold one of these architectural masterpieces.

A certain minimum number of soldiers were required to capture a castle. Some soldiers were expected to die during the attack. After capturing the castle, some soldiers were required to remain in the castle to defend it against attacks from another enemy. Of course, those numbers were different for different castles. Commanders of the armies were obliged to consider the number of soldiers required for victory.



For example, there are five castles in the region map shown in above Figure. The castle at the lower right requires at least 20 soldiers to wage a winning attack. None are expected to perish during the attack, and 10 soldiers must be left in the castle when the army moves on.

In this problem you must determine the minimum size of an army needed to capture and hold all the castles in a particular region. For reasons of security, there is exactly one (bi-directional) route between any pair of castles in the region. Moving into the neighbourhood of an uncaptured castle begins an attack on that castle. Any castle can serve as the first castle to be attacked, without regard for how the army got there. Once any castle has been captured, the requisite number of soldiers is left in the castle to defend it, and the remainder of the army moves on to do battle at another castle, if any remain uncaptured. The army may safely pass through the neighbourhood of a castle that it has already captured. But because of the potential for attacks, the army may traverse the route between a pair of castles no more than twice (that is, at most once in each direction).

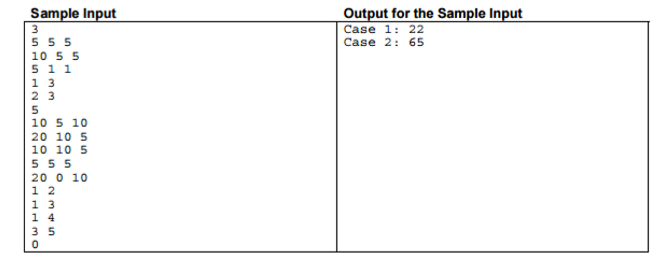
Input

The input contains multiple test cases corresponding to different regions. The description of the castles in each region occupies several lines. The first line contains an integer n ≤ 100 that is the number of castles in the region. Each of the next n lines contains three integers a, m, and g (1 ≤ a ≤ 1000, 0 ≤ m ≤ a, 1 ≤ g ≤ 1000), that give the minimum number of soldiers required to successfully attack and capture a particular castle, the number of soldiers that are expected to die during the attack, and the number of soldiers that must be left at the castle to defend it. The castles are numbered 1 to n, and the input lines describing them are given in increasing order of castle numbers. Each of the remaining n – 1 lines in a test case has two integers that specify the castle numbers of a pair of castles that are connected by a direct route.

A line containing 0 follows the description of the last region.

Output

For each test case, display the case number and the minimum number of soldiers in the army needed to conquer all the castles in the region. Follow the format shown in the sample output.



Program:

using System;

using System.Collections.Generic; // Using Namespaces

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assign5in13

{

class Program

{

static void Main(string[] args)

{

bool go = false, tr = false;

do

{

int N = 0;

Console.Write("Enter The Number of castles:\t");

// Getting the inputs from the console

N = Convert.ToInt32(Console.ReadLine());

int[,] a = new int[N, 3];

// Creating the array as per user input

int[,] b = new int[N, 3];

int cnt = 0, k = 0, r = 0;

// For loop to get the Soldiers Count for each castle

for (int j = 0; j < N; j++)

{

Console.Write("Enter the Soldiers by space seperator:\t");

string full = Console.ReadLine();

string[] parts = full.Split(' ');

a[j, 0] = Convert.ToInt32(parts[0]);

a[j, 1] = Convert.ToInt32(parts[1]);

a[j, 2] = Convert.ToInt32(parts[2]);

}

do

{

Console.Write("Enter the sequence by space seperator:\t");

// Enter the sequence of castle

string paths = Console.ReadLine();

string[] each = paths.Split(' ');

// Rearrange as per user input

for (int j = 0; j < each.Count(); j++) {

b[j, 0] = a[Convert.ToInt32(each[j]) - 1, 0];

b[j, 1] = a[Convert.ToInt32(each[j]) - 1, 1];

b[j, 2] = a[Convert.ToInt32(each[j]) - 1, 2];

}

// Finding the minimum soldiers for the sequence

for (int i = 0; i < N; i++)

{

// Check if the number of soldiers can win a castle

if (k >= b[i, 0] && k >= (b[i, 1] + b[i, 2]))

{

k = k - (b[i, 1] + b[i, 2]);

cnt++;

// After conquering all castle

if (cnt == N)

{

Console.WriteLine("Minimum required soldiers for conquering all castles in region is: {0}", r);

break;

}

}

// If the number of soldiers are not sufficient to win

else

{

r++; // increment the count

k = r; // Assign k

i = -1; // Initialize to -1 to make iteration from starting

cnt = 0; // Count is made as 0 so that it will start from first castle

}

}

Console.WriteLine("-----------------------------------");

Console.WriteLine("same region with different path press 1");

Console.WriteLine("continue for different region press 9");

Console.WriteLine("to exit press 0");

int s = Convert.ToInt32(Console.ReadLine());

if (s == 1)

go = true;

else if (s == 9)

{

tr = true;

go = false;

}

else if (s == 0)

{

go = false;

tr = false;

}

} while (go);

} while (tr);

Console.ReadKey();

}

}

}

Algorithm:

1. Get the number of Castles in the region.
2. Create a 2D array which holds the values for soldier’s count that is required to win, perish and defend.
3. Run a FOR loop to populate the array.
4. Get the sequence in a separate array and rearrange the 2D array for the specific sequence.
5. Run a FOR loop with limit as Castle count for determining the minimum soldiers required for conquering all the castles.
6. If the number is matching the user input for the castle to be conquered increment the count so that it accounts that the castle is won and if the count reaches the number of castle then print the soldiers required to win all castle.
7. Else increment the soldiers so that it will satisfy the required count ot conquer.
8. Reassign loop integer to -1 so the loop starts from first with the new soldiers count.
9. Build and Run

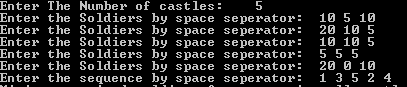
Inputs:

Required Inputs are

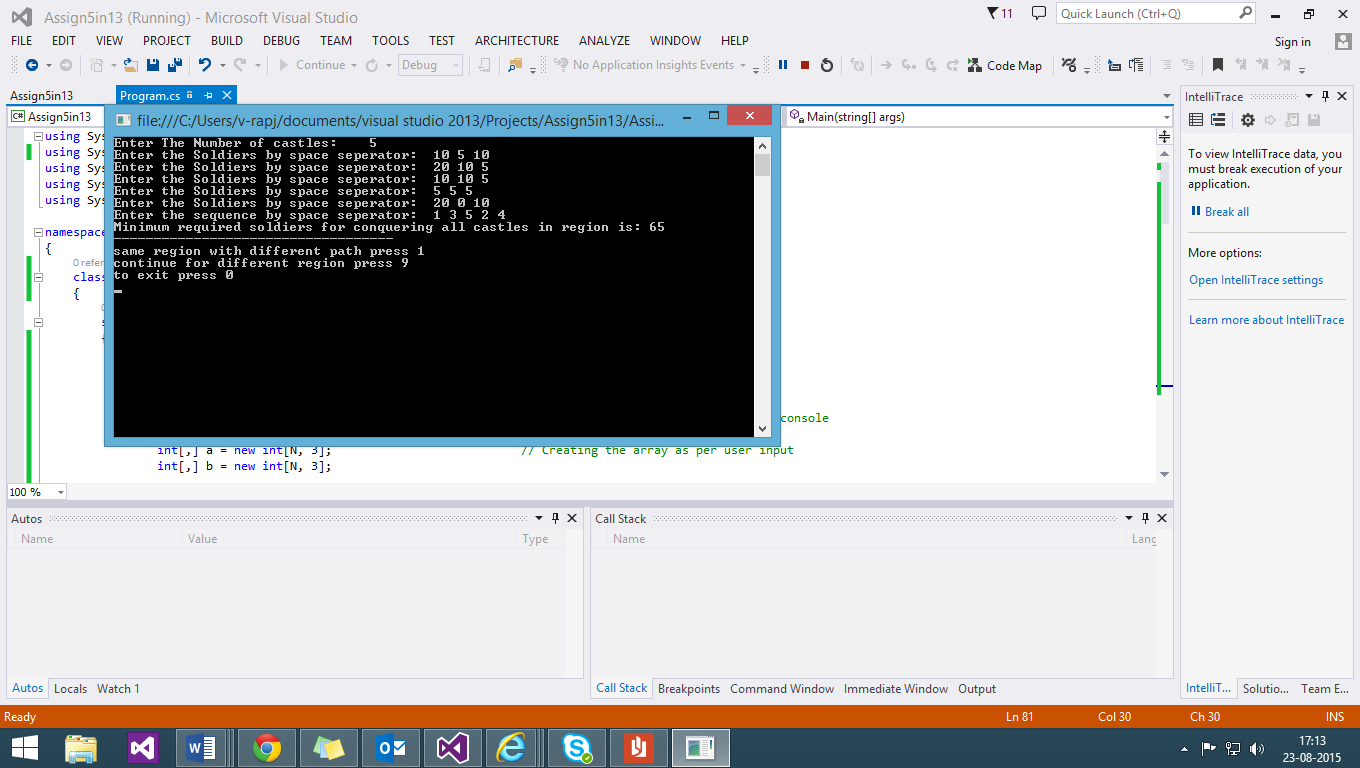
1. Number of Castle : N
2. Number of Soldiers required to win castle j : a[j,0]
3. Number of Soldiers die on the process : a[j,1]
4. Number of Soldiers required to defend : a[j,2]
5. Sequence of conquering the Castle : each[]

* Press 1 to continue with different sequence
* Press 9 to continue with different region
* Press 0 to exit

Test Case:



Output:



With Regards,

Anita, Harsh and Rajeshwaran