using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using UnityEngine;

using UnityEngine.AI;

[RequireComponent(typeof(UnityEngine.AI.NavMeshAgent))]

public class NavAgentExample : MonoBehaviour {

//public AIWaypointNetwork WaypointNetwork = null;

//public int CurrentIndex = 0;

//public bool HasPath = false;

//public bool PathPending = false;

//public NavMeshPathStatus PathStatus = NavMeshPathStatus.PathInvalid;

public static List<Transform> list = new List<Transform>();

public static int id = -1;

public static int did;

public static int exit = 0;

private UnityEngine.AI.NavMeshAgent \_navAgent = null;

// Use this for initialization

void Start () {

\_navAgent = GetComponent<UnityEngine.AI.NavMeshAgent>();

//if (WaypointNetwork == null) return;

//SetNextDestination(false);

}

public static void calfscore( Waypoint source, Waypoint child)

{

double f = Math.Sqrt(Math.Pow(source.x - child.x, 2) + Math.Pow(source.y - child.y, 2));

Waypoint destination = list.FirstOrDefault(a => a.id == did);

if (source.id == 0)

{

child.fscore = f;

child.gscore = calgscore(child, destination);

child.hscore = child.fscore + child.gscore;

child.fparent = source;

}

else if (child.fscore == -1)

{

child.fscore = f + source.fscore;

//child.gscore = calgscore(child, destination);

child.hscore = child.fscore + child.gscore;

child.fparent = source;

}

else

{

double fscore = f + source.fscore;

if (child.hscore > fscore + child.gscore)

{

child.fscore = fscore;

child.hscore = child.fscore + child.gscore;

child.fparent = source;

}

}

}

public static double calgscore(Waypoint child, Waypoint destination)

{

double g = Math.Sqrt(Math.Pow(destination.x - child.x, 2) + Math.Pow(destination.y - child.y, 2));

return g;

}

//hscore calculation

public static void calhscore(Waypoint source)

{

int i = 0;

int length = source.child.Count;

try

{

while (i <= length - 1)

{

foreach (Waypoint child in source.child)

{

calfscore(source, child);

calhscore(child);

}

i++;

}

}

catch (Exception e) { }

}

/\*public static void Addwaypoint(Waypoint w)

{

//Console.WriteLine("Enter id");

//int id = Convert.ToInt32(Console.ReadLine());

if (exit == 1)

{

did = 6;

}

//Console.WriteLine("Enter the x coordinates of the node");

int x = w.x;

//Console.WriteLine("Enter the y coordinates of the node");

int y = w.y;

Waypoint obj1 = list.FirstOrDefault(a => a.id == id);

if (obj1 == null)

{

obj1 = new Waypoint(id, x, y, new List<Waypoint>(), new List<Waypoint>());

list.Add(obj1);

//Console.WriteLine("Enter no of child");

//int child = Convert.ToInt32(Console.ReadLine());

//List<node> childnodes = new List<node>();

for (int i = 0; i < w.child.Count; i++)

{

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

node obj = list.FirstOrDefault(a => a.id == index);

if (obj == null)

{

// Console.WriteLine("node doesnot exist... adding new node");

obj = new node(index, -1, -1, new List<node>(), new List<node>());

list.Add(obj);

}

obj1.addchild(obj);

obj.addparent(obj1);

}

}

else

{

Console.WriteLine("Enter no of child");

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

List<node> childnodes = new List<node>();

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

{

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

node obj = list.FirstOrDefault(a => a.id == index);

if (obj == null)

{

// Console.WriteLine("node doesnot exist... adding new node");

obj = new node(index, -1, -1, new List<node>(), new List<node>());

list.Add(obj);

}

obj1.addchild(obj);

Console.WriteLine("node added");

}

obj1.x = x;

obj1.y = y;

foreach (node n in obj1.child)

{

n.parent.Add(obj1);

}

}

}\*/

static void Main(string[] args)

{

//List<Waypoint> finallist = new List<Waypoint>();

Waypoint source=

Console.WriteLine("For source:");

Addwaypoint();

while (exit != 1)

{

Addwaypoint();

Console.WriteLine("Press 0 for adding more node or press 1 for destination node");

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

}

Console.WriteLine("For destination");

addnode();

node source = list[0];

if (source != null)

calhscore(source);

node des = list.FirstOrDefault(a => a.id == did);

List<node> closedlist = new List<node>();

try

{

while (des.fparent.id != -1)

{

closedlist.Add(des);

des = des.fparent;

}

}

catch (Exception e)

{

closedlist.Add(des);

Console.WriteLine("nodes successfully inserted in closed list");

}

Console.WriteLine("node in order to reach destination in shortest path");

for (int i = closedlist.Count - 1; i >= 0; i--)

{

Console.Write(closedlist[i].id + " ");

}

Console.ReadKey();

}

}

/\*void SetNextDestination(bool increment)

// Update is called once per frame

{

if (!WaypointNetwork) return;

int incStep = (increment) ? 1 : 0;

Transform nextWaypointTransform = null;

int nextWaypoint = (CurrentIndex + incStep >= WaypointNetwork.waypoints.Count) ? 0 : CurrentIndex + incStep;

nextWaypointTransform = WaypointNetwork.waypoints[nextWaypoint];

if(nextWaypointTransform!=null)

{

CurrentIndex = nextWaypoint;

\_navAgent.destination = nextWaypointTransform.position;

return;

}

CurrentIndex++;

}\*/

/\*void Update () {

HasPath = \_navAgent.hasPath;

PathPending = \_navAgent.pathPending;

PathStatus = \_navAgent.pathStatus;

if ((!HasPath && !PathPending)/\*||(PathStatus=NavMeshPathStatus.PathInvalid))

{

SetNextDestination(true);

}

else

if (\_navAgent.isPathStale)

SetNextDestination(false);

}

}\*/

//Here instead of using arrow key u need to get the next point (Vector3 from A\*)

// and set the position of the transform using that value

//And u can use the console as debugger just as i did to see if the control is going

//to the code. Do u have any queries on how to use console etc?

//to see if the portion of the code is getting executed or not, we just type debug.log right??

//basically debug.log is similar to printf but it shows in the debugger (console)

//So yes u can use Debug.log() to see the output in console. Can u see the 2 debug logs

//i added in console?

//yes, i can see

//So now i need to remove main function and make some other function and check if it works or not

//First remove static keyword from the public variables. Then change main to some otehr function name.

//Even after if it doesnt work it means that there are some bugs in your code.

//But by putting Debug logs u can figure out where the bugs are. Makes sense?

//yes. I will check and get back to u as soon as possible.

//somehow have to finish it by tomorrow.

//No issues. U can try and if u dont get feel free to ping me.

//cool.Thank u so much

//NP. but in presentation explain about A\* and unity. It will impress the panel

// because the projects that other groups are doing are mostly straight forward and

//very common.

//Oh sure.

//Cool then. Bye

//ya bye