

# [Using return afstandtotaal; } public static float puntafstand(puntentijd](https://assignbuster.com/using-return-afstandtotaal-public-static-float-puntafstandpuntentijd/)

using System; using System. Collections. Generic; using System. Linq; using System. Text; using Android. App; using Android. Content; using Android. OS; using Android. Runtime; using Android. Views; using Android. Widget; using Android. Graphics; namespace RunningApp{    class Analyse    {        public static string Tracknaarstring(List track)        {            string trackstringres = “”;            int t = 0;            foreach (PuntEnTijd pt in track)            {                t++;                string voegtoe = “”;                voegtoe += pt. punt. X. ToString() + “?” + pt. punt. Y. ToString() + “?” + pt. tijd. ToString() + “?” + pt. pauze. ToString();                if (t < track. Count)                {                    voegtoe += "|";                }                trackstringres += voegtoe;            }            return trackstringres;        }            public static List Stringnaartrack(string trackstring)        {            List trackfake = new List();            string punten = trackstring. Split('|');            foreach (string punt in punten)            {                string puntdata = punt. Split('?');                PointF p = new PointF();                DateTime t;                p. X = float. Parse(puntdata0);                p. Y = float. Parse(puntdata1);                t = DateTime. Parse(puntdata2);                bool pz = false;                PuntEnTijd nieuwpunt = new PuntEnTijd(p, t, pz);                trackfake. Add(nieuwpunt);            }            return trackfake;        }        public static float Track\_Total\_Distance(List track)        {            PointF p = new PointF(0, 0);            DateTime t = DateTime. Now;            bool pz = false;            PuntEnTijd puntoud = new PuntEnTijd(p, t, pz);            PuntEnTijd puntnieuw;            int teller = 0;            float afstandtotaal = 0f;            foreach (PuntEnTijd punt in track)            {                puntnieuw = punt;                if (teller == 0)                {                }                else                {                    if (puntoud. pauze == true)                    {                    }                    else                    {                        if (puntnieuw. pauze == true)                        {                        }                        else                        {                            afstandtotaal += PuntAfstand(puntnieuw, puntoud);                        }                    }                }                puntoud = puntnieuw;                teller++;            }            return afstandtotaal;        }        public static float PuntAfstand(PuntEnTijd eerstepunt, PuntEnTijd tweedepunt)         {            float lengteX = Math. Abs(eerstepunt. punt. X - tweedepunt. punt. X);            float lengteY = Math. Abs(eerstepunt. punt. Y - tweedepunt. punt. Y);            return lengteX + lengteY;        }    }    public static float PuntTijdVerschil(knooppunt een, knooppunt twee)    {        return Math. Abs(twee. timesincestart - een. timesincestart);    }}