**Plan van Aanpak Concurrency pract2**

Wat?

Sockets maken

* Thread voor elke socket
  + Interactie met console
  + Aannemen inkomende verbindingen
  + Communicatie met buurman (voor elke buur)

Invoer Netwerk

* Toon routing table – R
  + Laat huidige table Nbu(v) zien (gegeven aan thread u)
* Stuur Bericht – B ***poortnummer******string***
  + Verstuurt ***string*** naar ***poortnummer*** waarna ***poortnummer*** dit print
  + Bij onbekend poortnummer: foutmelding
* Maak verbinding – C ***poortnummer***
  + maakt ***poortnummer*** directe buurman van u (thread waarin dit bericht komt)
  + Wordt alleen op u gegeven, niet ook op v als Nbu(v)
* verbreek verbinding – D ***poortnummer***
  + verbreekt ***poortnummer*** directe buurman van u (thread waarin dit bericht komt)
  + Wordt alleen op u gegeven, niet ook op v als Nbu(v)
  + Bij ***poortnummer*** geen buur: foutmelding
  + LET OP: partitie kan ontstaan

Uitvoer

* Bijwerken afstandsschatting
  + *Afstand naar v is nu d via h*
* Bij ontstaan van directe verbinding met proces op poort p
  + *Verbonden: p*
* Bij het verbreken van een directe verbinding met het process op poort p:
  + *Verbroken: p •*
* Als het process op poort p door een netwerkpartitie onbereikbaar is geworden:
  + *Onbereikbaar: p •*
* Indien bij een B- of D-commando een ongeldig poortnummer p wordt opgegeven:
  + *Poort p is niet bekend*
* Als een bericht wordt doorgestuurd:
  + *Bericht voor v doorgestuurd naar h*
* Als een bericht voor het process zelf wordt ontvangen toon dit bericht dan verbatim op deconsole.
* Als je extra debug-informatie wil printen laat dit dan voorgaan door //:
  + //Deze regel wordtdoor TomJudge genegeerd, lucht hier gerust je hart!

Nog een aantal extra aanwijzingen en belangrijke tips:

• Het afhandelen van inkomende connecties dient in aparte threads te gebeuren. Elk proces heeft daarom voor elke buurman een aparte thread die inkomende berichten afhandelt.

• Vergeet niet om toegang tot gedeelde data (met name: routing table) op de juiste manier te synchroniseren. Je mag voor deze opdracht gebruik maken van de ingebouwde locks in C#, maar niet van objecten uit bijvoorbeeld de System.Collections.Concurrent namespace. Als je zulk soort objecten wilt gebruiken moet je die zelf bouwen.

• Het is mogelijk dat een proces probeert te verbinden met een proces dat nog niet opgestart is. In dat geval moet je de opgeleverde exception afvangen, een aantal miliseconden slapen, en nog een keer proberen.

• Onnodig gebruik van spinlocks wordt zwaar gestraft in het cijfer. Een netwerk dat opgestart is zou, indien idle, een zeer klein percentage van de CPU moeten gebruiken. Programma’s die +/- 100% CPU gebruiken in idle stand kunnen en zullen niet nagekeken worden. Als je programma zoveel CPU gebruikt dat het testen in TomJudge je computer doet vastlopen, kunnen wij er geen cijfer aan geven.

• Bij het verbreken van verbindingen kan er een netwerkpartitie onstaan. Hierdoor komt het Netchange-algoritme in een oneindige loop waarbij de afstandschatting steeds omhoog gaat. Dit moet je afvangen zodat de loop stopt. (Hint: je programma hoeft een netwerk met maximaal 20 clients aan te kunnen. Het kan ook nog slimmer!)