Udfordringer - Benjamin, Martin, Michael og Jannik

Dag 1:

- At integrere og gøre brug af Telnet.cs classen
    - Der var mange unødige funktioner i den class der allerede var lavet til at etablere en Telnet-forbindelse, til vores formål
- Vi var ikke sikre på vi havde en forbindelse til routeren, så vi tilføjede nogle Console.WriteLine's, der giver os besked hvis forbindelse er blevet etableret korrekt.
- Finjusteringer af kode, fjerne det vi ikke gjorde brug af og små 'snippets' fra funktioner der var overflødige

Dag 2:

- Opbygge en menu med flere valgmuligheder, der vælger punkt alt efter brugerens valg
- Lave programmet mere dynamisk (fx input af ip og password)
    - Ideen er, at man kan bruge programmet på flere enheder med denne metode
- Dynamiske metoder
    - Fx selv indtaste hostname, motd, og password efter ønske
- Vi stødte på et problem, da vi skulle bruge en funktion der ændrede hostname på enheden
    - Problemet var, at den funktion der allerede eksisterede (i Telnet.cs) tjekkede på hostname, og når det var ændret til noget nyt, kørte i et uendeligt loop
    - Vi kom frem til en løsning, der involverede at duplikere den allerede eksisterende funktion, og lave nogle få ændringer (se nedenunder)

Nedenfor ses et udklip kode med pil ud for ændringer der er foretaget.

```
        public List<String> CiscoCommand(string Command)
        {
            String Result;

            List<String> ResultList = new List<String>();
>            WriteLine(Command);
            Result = ReadUntil("(" + hostname + "#|" + hostname + "\\(config.*\\)#)");
            String[] ResultArray = Result.Split('\r');
            //Remove newline and replace tabs with space
            for (int i = 0; i < ResultArray.Length; i++)
            {
                ResultArray[i] = Regex.Replace(ResultArray[i], @"\n|\r", "");
              // Remove \r and \n
                ResultArray[i] = Regex.Replace(ResultArray[i], @"\t+", " ");
              // Replace one or more tabs with one space
                ResultList.Add(ResultArray[i]);
            }
            return ResultList;
        }
```

Denne funktion blev duplikeret; her er løsningen

```
        public List<String> CiscoHostname(string newname)
        {
            String Result;

            List<String> ResultList = new List<String>();
>            WriteLine("hostname " + newname);
>            hostname = newname;
            Result = ReadUntil("(" + hostname + "#|" + hostname + "\\(config.*\\)#)");
            String[] ResultArray = Result.Split('\r');
            //Remove newline and replace tabs with space
            for (int i = 0; i < ResultArray.Length; i++)
            {
                ResultArray[i] = Regex.Replace(ResultArray[i], @"\n|\r", "");
               // Remove \r and \n
                ResultArray[i] = Regex.Replace(ResultArray[i], @"\t+", " ");
               // Replace one or more tabs with one space
                ResultList.Add(ResultArray[i]);
            }
            return ResultList;
        }
```

WriteLine blev ændret, og “hostname” blev tilføjet - dette er kommandoen til at ændre hostname på enheden.
Derefter blev den globale variabel ‘hostname’ re-assigned til ‘newname’ - denne string bliver parsed til funktionen, inde fra Program.cs (fra user-input)
På denne måde bliver hostname ændret korrekt (før ledte den efter det forrige hostname og kørte aldrig videre i koden)

Det har til tider været en udfordrende opgave, mest på det aspekt med at finde rundt i den allerede eksisterende kode.

Vi lærte at i stedet for at kigge på problemet som en helhed, kan det være vigtigt at prøve at tage det skridt for skridt, og løse det på den måde. Ellers kan det hurtigt blive meget uoverskueligt.