Mål

Målet med detta program var framför allt att läsa av JSON data och extrahera intressant information för att sedan jämföra informationen och skriva ut det till en terminal för avläsning. Från början var tanken att enligt instruktionerna skriva detta script i de önskade språken (bash eller Go). Efter ett samtal med min kontakt blev instruktionerna att inte skriva programmet i bash, utan att i så fall skriva det i Go. Detta blev svårare än förväntat att lära sig tillräckligt mycket på kort tid så i slutändan blev programmeringsspråket som används Python. JSON data består av “keys” och ”values” för att spara data, dessa kallas ibland för key/value par. Båda dessa innehåller information som Python kommer att analysera, “keys” är ungefär som en variabel som sparar ett värde (“values”) i en JSON struktur.

Tillvägagångssätt

Baserat på instruktionerna jag fick ta del av blev den första delen för mig att generera mer exempel data så att programmet har mer att läsa av. Den första delen av programmet går alltså ut på att med samma format som det givna exemplet skriva en fil med fem exempel objekt som sedan kommer att läsas från samma fil och användas för den andra delen av programmet. Efter att en fil har skapats kommer programmet sedan öppna och läsa datan ur filen och göra om så att python lätt kan använda den data.

När JSON objekten har analyserats går programmet igenom alla objekt och plockar ut de viktigaste delarna. Objekten består av “timestamp” och “message”. I “timestamp” finns bara ett värde i form av en tidsstämpel och i “message” finns flera keys som behöver analyseras vidare. För varje objekt sparar programmet “timestamp” i en variabel för att senare kunna jämföras i nästa del. I “message” finns då flera keys, där ibland “body” som har två keys, “c” och “body”. I “body” hittar vi “dp” vilket innehåller den information vi vill åt. I “dp” finns “IDT” och i varje IDT finns det ännu en tidsstämpel, den tidsstämpeln kommer senare att jämföras med den första tidsstämpeln. Först skapas en lista på alla IDT för att lättare kunna komma åt alla tidsstämplar som finns i vardera, sen kollas alla IDT igenom och tittar om tidsstämpeln i IDT-objektet är mer än en timme senare än den ursprungliga stämpeln. Om det är mer än en timme, skrivs objektets ordningsnummer, IDT, start tidsstämpeln och slut tidsstämpeln.

Svårigheter

De största svårigheterna jag har när det kommer till att utföra kodprojekt är planering. Att planera ett program där jag själv hade svårt att förstå instruktioner till, samt i ett språk jag aldrig har använt förut var en stor utmaning. Att välja i slutändan att använda mig av ett språk som jag är mer bekväm i var ett svårt beslut eftersom att jag alltid vill visa att jag vill och kan lära mig nya saker. Den här gången blev det dock lite svårare att hinna med att göra programmet utan kunskaper i Go. Att skriva programmet i ett språk som jag är bekväm i gjorde att själva kodningen gick smidigare eftersom att jag inte har behövt söka på alltför mycket information och

hjälp, vilket gör att det kunde gå fortare. Förutom det var det inte så mycket problem i utförandet av koden, vilket jag är tacksam för.

Det svåraste jag tyckte när jag studerade var att göra upp planer och rita upp designer av program, jag har alltid haft lättare att “bara göra”. Det gjorde att diagrammet jag skulle göra på “draw.io” blev den största utmaningen i planeringen. Jag har alltid haft svårt med pseudokod, vilket gjorde att relationen mellan de olika delarna i programmet blev svårt att illustrera i ett diagram. Oavsett det så blev jag nöjd med resultatet.

Resultat

I slutändan tyckte jag att det var en riktigt kul utmaning, både med utförandet och planeringen. Jag är nöjd med resultatet och hoppas att det inte har blivit någon brist från min sida att förstå hur uppgiften skulle utföras och vad för resultat det skulle medföra.