Seminarie 4

Uppgift 1

Vi valde att göra nätverket som uppgiften rekommenderade, alltså en konverterare mellan celsius och farenheit. För konverteringen använder vi oss av ekvationen *f = c * 1,8 + 32*.

Det första vi gjorde var att testa Adder och Multiplier. Vi baserade testerna på de testfunktioner som redan fanns i koden och insåg då att *new_value()* endast kunde hantera fall där a och b hade givna värden medan resultatet out var tomt. Vi lade till if-satser för att också kunna hantera fall där det istället var a eller b som inte hade några värden. Efter detta fungerade våra tester, även efter att vi utökat dem för att testa alla tre variabler.

Konverteringsfunktionen gav inga riktiga problem under tiden vi byggde på den. Vi skapde c, f och out precis som a/b/out tidigare, men också de två konstanterna k1 (1,8) och k2 (32) från ekvationen. Sedan följer vi ekvationens naturliga gång (*(c * k1) + k2 = f*) och får ett svar (ibland med väldigt många decimaler, men vi hade några små problem med avrundning så vi lät det vara).

Uppgift 2

Uppgift 2 tog överlägset längst tid att få till och var en rejäl utmaning i att få en förståelse för kod man inte själv har skrivit. Det började bra med att vi jagade error-meddelanden för att rota ut diverse småfel (som en felskriven *require*), men vi fastnade sedan på ett felmeddelande kring en name-metod som inte fanns vilket ledde till att vi spenderade mycket tid med att försöka ”fixa” *replace_conn()* genom att ändra på villkoren och hur *.name* respektive *.out* användes när felet egentligen satt i *get_connector()* och *get_connectors()*.

Till slut insåg vi vad problemet var och åtgärdade det bara för att sen stöta på ett annat fel som tog en lång tid att lösa, nämligen att fel värden verkade användas i multiplikationsdelen av konverteringen. Till exempel användes f istället för f-32 i *f-32 * 5 == 9c*. Detta ledde oss till slut till match-reglerna där vi skapade ett par nya klasser i *constraint-networks.rb* specifikt för subtraktion och dividering och använde dem i match-reglerna på de platser där *adder* och *multiplier* tidigare användes för de operatorerna, vilket löste problemet.

Vi tror inte att detta nödvändigtvis var den lösning som det var tänkt att vi skulle använda, och vi fick intrycket att man egentligen skulle ha modifierat hur *adder* och *multiplier* används för subtraktion och dividering, men vår experimentering gav inga resultat så vi använde denna lösningen istället.

Debug-utskriften i vårt program skiljer sig till viss del från det som står som exempel längst ner i filen, men det är bara namn och ordning på vissa connectors och uträkningar som är annorlunda, och svaren blir fortfarande korrekt