Utvecklarblogg - Seminarium 4

Grupp B6 - guser766 | hadan326 | vinah331

Tolkning av uppgifter och förklaring av lösningar

Uppgift 1

Vi började med att läsa igenom filen följt av en felsökning. Problemet var att new_value funktionen saknade alternativ för omvandling av annat än a och b till c. Lösningen var helt enkelt att skapa två elsif-satser där enda skillnaden från if-satsen var att a respektive b bytte plats med c.

```
elsif [out,b].include?(connector) and out.has_value? and b.has_value? and
      (not a.has_value?) then
.# När a saknar värde i ekvationen a + b = out
.
# a = out - b
     val=out.value.send(inverse_op, b.value)  # omvända + till - för att skapa ekvationen a = out - b
.
.



elsif [out,a].include?(connector) and out.has_value? and a.has_value? and
      (not b.has_value?) then

.# När b saknar värde i ekvationen a + b = out
.
# b = out - a
     val=out.value.send(inverse_op, b.value)  # omvända + till - för att skapa ekvationen b = out - a
.
.
.
```

def celsius2fahrenheit()

Vi tolkade den här delen så att funktionen celsius2fahrenheit ska returnera två objekt som representerar celsius och fahrenheit där de två objekten är kopplade till varandra.

Vi utgick från följande formellen:

f = c * 1.8 + 32

För att sätta upp ett constraint-nätverk använde vi oss av fem connectors:

```
c = Connector.new("c")
f = Connector.new("f")
x = Connector.new("1.8", 1.8)
y = Connector.new("y")
z = Connector.new("32", 32)
```

Connector y innehåller outputen på (f -32).
Sedan ska man definiera ett korrekt matematisk nätverk för att få det att fungera. Som vi ser i formeln så krävs det en Multiplier för “c * 1.8” och en Adder för “y + 32”. Därför skriver vi så här (det som kommer efter pilarna är tolkningen av raden).

```
Multiplier.new(c, x, y) #-----> c * 1.8 = y
                                          #-----> f = c * 1.8 + 32
Adder.new(y, z, f)      #-----> f = y + 32
```

Till slut när vi har skapat detta och kopplat våra connectors så returnerar vi dem med:
return c, f

Vi har testat att funktionen kan omvandla celsius till fahrenheit och det visade sig att den gör den utan några konstigheter.

Uppgift 2

Vi började även här med att granska all kod i ett försök att förstå hur programmet är tänkt att fungera. Uppgiften uppfattades som något otydlig och vi visste inte vad vi fick och inte fick ändra. Vi insåg till slut att get_connector ännu inte implementerats så vi startade där. Det ledde till granskning av diverse outputs. Vilka klasser hanteras och hur skiljer vi dessa åt? En enkel if sats löste detta. Även i get_connectors behövde denna distinktion göras.

Efter dessa åtgärder körde vi fast rejält. Vi fick somliga ekvationer att fungera genom att returnera conn_c i adder-matchningen vilket förvirrade oss. Hur viktigt var det att skicka vidare constraints? Efter samtal med handledare förstod vi att det inte var rätt väg att gå. Hanteringen av subtraktion och division skulle bli besvärlig med vår lösning.

Den optimala lösningen visade sig vara att skapa två nya constraints, subtractor och divider. Det kändes lite “fuskigt” när det smidigaste lösningen tog tio sekunder att implementera. Var allt en kuggfråga? Samtidigt hade vi gått miste om mycket förståelse om vi vetat om det från början.

Vid testningen fokuserade vi på att testa de fyra matematiska uttrycken.