Linus Frisk
010809-2933
4/17/2025

Uppgift 7

Test

Uppgiften gick ut på att man skulle bygga en testfil till en föregående uppgift electrotest-standalone.c. Vilket var en fil som beräknade elektriska komponenter i ett system.
Grunduppgiften här var att se på denna filen som en “black box” och skriva tester med hjälp av CUnit vilket är ett ramverk som innehåller testfunktioner till C programmering.

Dem två huvudmålen var att funktionerna returnerar rätt värde för korrekt indata samt felaktig indata skulle hanteras på ett säkert sätt, det vill säga utan några krascher i programmets körning.

För att göra detta började jag med att skapa en testfil som heter test_file.c där jag följde exempelfilen ifrån cunit.sourceforge [1]. Där jag bland annat la in typiska saker som **init_suite1 clean_suite1** och vanlig kod för att initialisera CUnits test registry. Allt detta går att läsa i den bifogade filen i .tar.gz mappen.

Därefter hade vi tre olika funktioner att prova (plus en extra som var en extrauppgift). Dessa funktionerna var.
calc_power_r(volt, resistance)
calc_power_i(volt, current)
calc_resistance(count, conn, array) (extrauppgift)
e_resistance(orig_resistance, res_array)
När jag skrev testerna använda jag mig också utav en annan hemsida ifrån cunit, vilket gav mig verktyg för vilka typer av CU_ASSERT jag skulle använda (i detta fallet DOUBLE_EQUAL) [2].

Nedan kommer alla former av felaktig indata samt korrekt indata.
Test av felaktig indata

Funktion	Feltyp	Förväntad hantering
calc_power_r	resistance = 0	Returnerar -1.0
calc_resistance	count = -1	Returnerar -1.0
calc_resistance	array = NULL	Returnerar -1.0
calc_resistance	conn = ‘X’	Returnerar -1.0
e_resistance	orig_resistanec = -10	Returner antal >= 0, kraschar inte.

Linus Frisk
010809-2933
4/17/2025
Test av korrekt värde

Funktion	Indata	Förväntat resultat
calc_power_r(10, 5)	Volt = 10, Resistance = 5	20.0
calc_power_i(10, 2)	Volt = 10, Current = 2	20.0
calc_resistance([10,20,30], 'S')	Seriekoppling	60.0
calc_resistance([10,20],'P')	Paralellkoppling	6.666
e_resistane(1000)	Orignalresistor = 1000	1000 ohm, 3 motstånd

```
passed

Run Summary:    Type  Total    Ran Passed Failed Inactive
              suites      1      1    n/a      0        0
               tests      4      4      4      0        0
             asserts     11     11     11      0      n/a

Elapsed time =    0.004 seconds
```

På bilden kan vi då se att alla fyra tester har körts och dem alla har passerat och att alla de olika indata(asserts) har skickats in korrekt och även där passerats

När det kommer till extrauppgiften så användas samma typ av testfunktioner till just denna. Bara att det blev lite mer arbete med arrays och hur man fick tänka till på hur man skrev dessa för att få fel indata samt rätt indata.
Själva "algoritmen" för alla är egentligen bara att man kör programmet som sparar värdet till en **result** och därefter har man en **CU_ASSERT_DOUBLE_EQUAL** som kollar om resultatet är likt det förväntade resultatet och gör en utskrift beroende på hur dem stämmer överens.

Sjukt roligt uppgift, kul att få peta med en gammal uppgift och skriva tester.
https://cunit.sourceforge.net/example.html [1]
https://cunit.sourceforge.net/doc/writing_tests.html [2]