## OPDRACHT 2. CPU-LOAD

In deze opdracht onderzoek je hoe processen de CPU load beïnvloeden. Het operating systeem zorgt namelijk dat de processortijd op een goede manier verdeeld wordt over de processen die draaien. Later komen we terug op hoe die scheduling werkt.

### 1. Voorbereidingen

Installeer het programma `KSysGuard` (KDE Task Manager and Performance Monitor) via `het K Menu`. Dit is.een Linux grafisch programma om de CPU load te zien; vergelijkbaar met de Windows Task Manager.

Download en unzip de file *IPC32_ipc.zip*. Hierin vindt je een programma `dutycycle.c`, een testprogramma wat we gaan gebruiken om de CPU te belasten.
Vertaal het programma met het volgende commando:

```
gcc dutycycle.c -o dutycycle -lrt
```

### 2. Taken

Onderzoek wat de invloed is van een enkel process (of van meerdere processen parallel) op de CPU-load. Onderzoek specifiek het volgende:

- het process voert eigen code uit
- het process voert system calls uit
- het process slaapt voor een groot gedeelte van de tijd
- meerdere processen geven elkaar steeds de beurt.

Het Kubuntu/Lubuntu VMware image is opgezet voor 1 processor core. Meerdere processen moeten dan achter elkaar op dezelfde core draaien. Wellicht draai je zelf native Linux op een multiprocessor core; dan zul je waarschijnlijk nog interessanter gedrag zien.

Het C programma `dutycycle.c` biedt de nodige mogelijkheden om de CPU te belasten. In de opdrachten wordt je gevraagd om meerdere shells naast elkaar op te starten. Start in elke shell dan het programma op (met bijv. deze parameters):

```
./dutycycle 1000 0 0 0
```

Start daarnaast ook `KSysGuard` op. Maak aanpassingen zodat KSysGuard de juiste informatie over het systeem laat zien. Creeer daartoe een nieuw tabblad via *File -> New Tab* en voeg dan de volgende sensors toe: *CPU Load -> System -> User Load & Nice Load & System Load*, en *CPU Load -> Context Switches*.

a. Start in een shell `dutycycle` met parameters 1000 0 0 0. En daarna een tweede `dutycycle` in een andere shell (met dezelfde parameters).
   Verklaar wat je ziet in de `CPU Load` en `Context Switch` schermpjes van `KSysGuard`.
   Stop de processen (met `Ctrl-C`).
b. Doe hetzelfde met de parameters
   0 1000 0 0
   0 0 1000 0
   0 0 0 1000
c. Wat betekenen de diverse parameters?
d. Start het programma ook op met een lagere prioriteit (aanwijzing: zie `nice`), of lage en hoge prioriteit naast elkaar.
   Beschrijf wat er gebeurt.
e. … (speel zelf met de parameters, en ga op jacht naar verrassende scenario's )
f. `dutycycle` print elke seconde 2 getallen.
   Wat betekenen deze getallen? Kijk hiervoor in de code naar de functie `another_cycle()`: wanneer wordt die aangeroepen, en wat doet deze functie met de variabele `counter`?

Tip: het is niet verboden om in de source code te kijken wat de parameters doen en wat de output betekent!

**3. Opleveren**

Je dient een document op te leveren waarin je de bovenstaande experimenten uitwerkt. Voeg screendumps toe om een en andere duidelijk te maken.