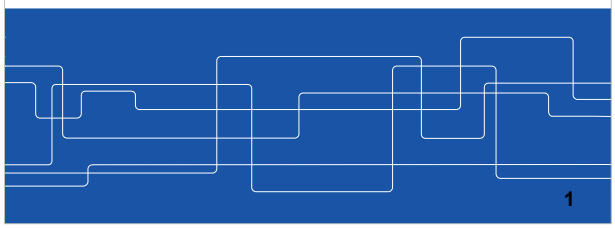
Föreläsning 11

Trådning



Trådning i Python

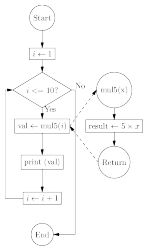
- I Python implementeras trådning av modulen `threading`
- Trådobjekt skapas av konstruktören `Thread ()`
- Den nya, oberoende, exekveringstråden startas genom att föräldertråden anropar metoden `start ()` på trådobjektet
- Den nya "barntråden" kör då trådobjektets metod `run ()`
- Barntråden upphör att existera då `run ()` returnerar
- Föräldratråden kan vänta på att en barntråd är färdig genom att anropa trådobjektets `join ()`

Exekveringstråden

- Ett datorprogram arbetar normalt i en serie steg: När ett steg är färdigt utförs nästa.
- Exempel:

```
def mul5 (x):
    return 5 * x

i = 1
while i <= 10:
    print (mul5 (i))
    i = i + 1
```



Trådning i Python

- Enklare variant (som använder den fördefinierade `run ()` i klassen `Thread`):
- Definiera en funktion, tex `foo ()`, som ska köras i oberoende tråd
- Skapa det nya trådobjektet genom:

```
nyTråd = Thread (target = foo)
```

- `run ()` kommer nu att anropa `foo ()`

Trådning

- Ibland vill man använda *fler* exekveringstrådar som löper parallellt. Det kallas trådning.
- Program för vissa problem blir enklare och tydligare med trådning eftersom en tråd kan sköta en uppgift.
- Trådning kan utnyttja hårdvaran bättre, tex därför att en processor kan låta varje kärna köra en tråd.
- Det finns andra former av parallellism, tex vektorparallellism (då många operationer utförs parallellt av samma instruktion) eller processparallellism (då en "tråd" körs som en självständigt program i en annan minnesrymd).