

Moduler, paket

När vi önskar använda oss av Pythons matematiska funktioner skriver vi:

```
import math
```

Vi har därmed tillgång till det som finns i *modulen* math. Exvis de matematiska funktionerna:

```
x = math.sqrt(7)
```

Eller så här:

```
import math as m  
x = m.sqrt(7)
```

Antag att vi skrivit en fil `temp.py` med kod (python-satser) och skall köra den filen. Då kallas modulen för `temp`. Den modul man startar programmet i kallas även för *main module*.

När man skriver ett lite större program som består av flera funktioner kan det vara opraktisk att ha alla funktioner i samma fil. Dessa funktioner kan dessutom vara bra att använda i andra program. Därför spar man dessa funktioner i separata filer, s.k. moduler.

Programmen som använder funktionerna importerar de moduler där dessa funktioner finns.

Python innehåller många färdiga moduler, exvis *math*, *random*, *graphics*

Istället för att importera en hel modul kan man importera ett funktionsnamn som finns i modulen, exvis:

```
from math import sqrt
```

Därmed kan man skriva:

```
z = sqrt(7)
```

Man kan importera alla funktioner från en modul:

```
from math import *
```

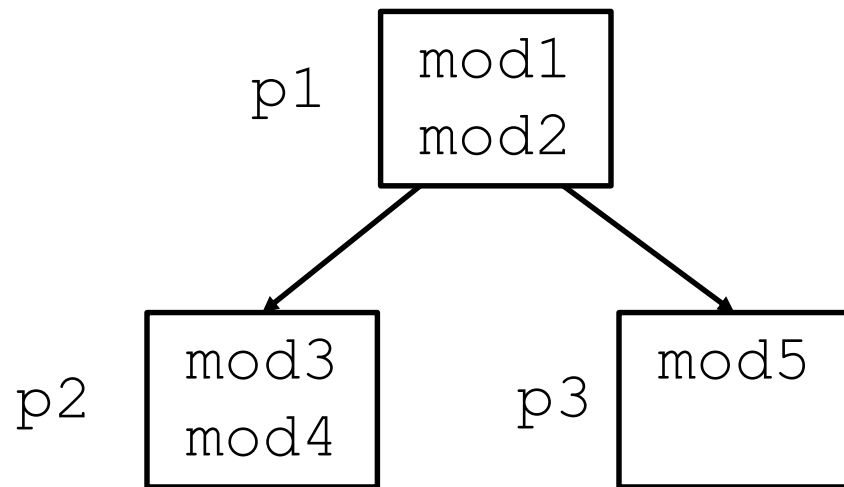
Därmed kan man skriva:

```
z = sqrt(7)
```

```
y = pow(2.3, 6)
```

Paket

För att hålla reda på moduler kan dessa organiseras i mappar och undermappar. Ett exempel: Paketet `p1` består av modulerna `mod1` och `mod2`. Paketet `p2` består av modulerna `mod3` och `mod4`. Paketet `p3` består av modulen `mod5`. I datorns filsystem är `p1` en mapp med filerna `mod1.py` och `mod2.py`. Mapparna `p2` resp `p3` är submappar till `p1`.



Tre exempel på anrop:

```
import p1.p2.mod3
p1.p2.mod3.fx() # Anropar fx i mod3
```

```
from p1.p3.mod5 import fy
fy() # Anropar fy i mod5
```

```
from p1.mod2 import fz
fz() # Anropar fz i mod2
```