Skolan för Datavetenskap och kommunikation

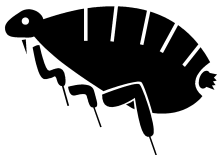
DD1312
Programmeringsteknik

Föreläsning 14

idag

- Buggar
- Formatering med f-strings
- Felhantering med try...except
- Kopiera en lista
- Kopiera en lista av objekt

olika typer av buggar



- Felavbrott (Exception) när programet körs
- Inget händer när du kör programmet
- Massor av text rinner över skärmen
- Programmet gör något annat än det du önskade
- Programmet gör rätt för vissa indata, men inte för andra

felavbrott

Lär dig tolka felutskrifter!

```
Traceback (most recent call last):
  File ”filmer.py", line 124, in <module>
    titta(listan)
  File ”filmer.py", line 102, in titta
    film.ny_visning(1)
TypeError: ny_visning() takes exactly 1 argument (2
   given)
```

tolkning

- Sista raden förklarar felet!

```
TypeError: ny_visning() takes exactly 1 argument (2 given)
```

- Raderna ovanför visar anropskedjan. Läs nerifrån och uppåt!

```
File ”filmer.py", line 102, in titta
    film.ny_visning(1)
```

- Det var i funktionen `titta()` som anropades på rad 124.

```
File ”filmer.py", line 124, in <module>
  titta(listan)
```

fixa felet

- Så här ser koden ut

```
def ny_visning(self):
    """ökar antal visningar"""
    self.visningar += 1
```

- Vi tar bort parametern i anropet:

```
film.ny_visning()
```

kontrollutskrifter

- Använd kontrollutskrifter för att hitta var i programmet felet uppstår.
- En kontrollutskrift är en vanlig print-sats, till exempel:

```
print("Klar med inläsningen")
```

- Du kan också skriva ut variabelvärden för att se hur dom ändras under körning.

```
print("x=", x)
```

- Eller stanna upp så här:

```
input("x=", x, "Tryck Enter")
```

Mittiprick-metoden

Anta att programmet hänger sig, men vi vet inte *var* i programmet det inträffar.

1. Lägg en kontrollutskrift i *början* och en i *slutet* av huvudprogrammet. Felet ligger däremellan.
2. Lägg in en kontrollutskrift *mitt i*. Om den kommer ut som den ska finns felet i andra halvan, annars i första.
3. Fortsätt tills du hittat felet!

```
input("Start")

for i in range(1,11):
    print(i/10)
print()

for i in range(1,11):
    print(i/10, end = "   ")
print("\n")

input("Efter for-slingorna")

x = 0
while x != 1:
    x += 0.1
    print(x)

x = 0
while round(x) != 1:
    x += 0.1
    print(x)

input("Slut")
```

Terminal/Power Shell

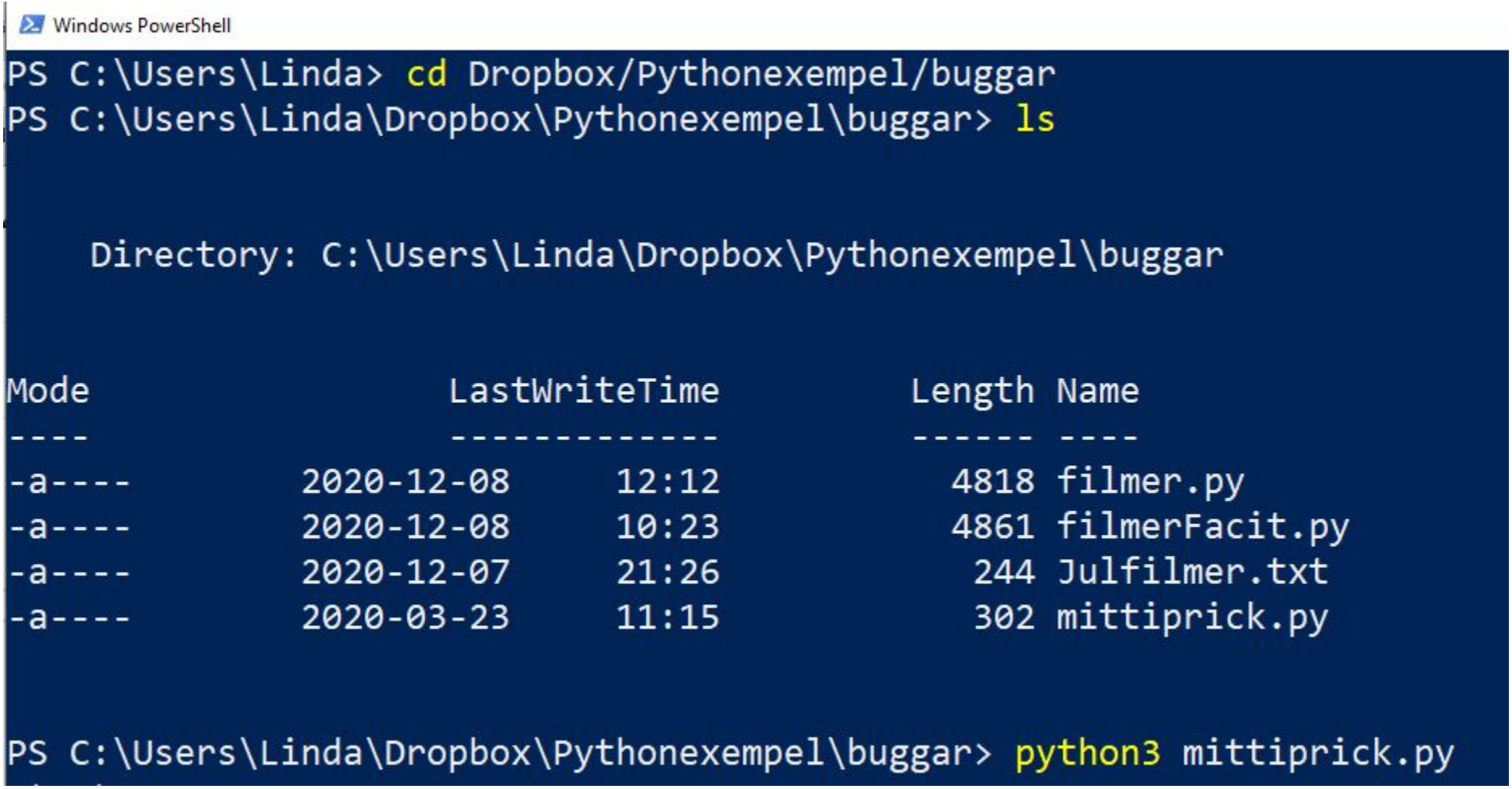
Din utvecklingsmiljö (*IDLE, PyCharm, PyDev, Spyder, Atom, VisualStudio, ...*)

kan ibland ge märkliga resultat. Prova att köra ditt program i

Terminal (Mac/Ubuntu)

eller

Power Shell (Windows)


Windows PowerShell
PS C:\Users\Linda> cd Dropbox/Pythonexempel/buggar
PS C:\Users\Linda\Dropbox\Pythonexempel\buggar> ls
Directory: C:\Users\Linda\Dropbox\Pythonexempel\buggar
Mode LastWriteTime Length Name
---- ------------- ------ ----
-a---- 2020-12-08 12:12 4818 filmer.py
-a---- 2020-12-08 10:23 4861 filmerFacit.py
-a---- 2020-12-07 21:26 244 Julfilmer.txt
-a---- 2020-03-23 11:15 302 mittiprick.py
PS C:\Users\Linda\Dropbox\Pythonexempel\buggar> python3 mittiprick.py

formaterad utskrift

(som chatGPT använder)

formaterad utskrift

Hur skriver man ut en tabell med raka kolumner?

Eller ett värde med två decimaler?

Ange *formatering* i print-satsen med en *f-sträng*

Skriv

```
print(f"Svaret = {x:pos}")
```

där

x är en variabel eller ett värde

pos är antal positioner

exempel

```
>>> pris = 99
>>> print(f"Den kostar {pris} kr")
Den kostar 99 kr
```

antal positioner

{x:4}	4 positioner	`_ _ 1 7`
{x:6.4}	6 positioner, 4 siffror	`_ 3 . 1 4 2`
{x:>7}	7 positioner, högerjusterat	`_ _ _ k a t t`
{x:<7}	7 positioner, vänsterjusterat	`k a t t _ _ _`

dokumentationen

ful tabell

```
for n in range(10):
    print(n, n**2, n**3, n**4)
```

```
0 0 0 0
1 1 1 1
2 4 8 16
3 9 27 81
4 16 64 256
5 25 125 625
6 36 216 1296
7 49 343 2401
8 64 512 4096
9 81 729 6561
```

finare tabell

```
for n in range(10):
    print(f"{n} {n**2:6} {n**3:6} {n**4:6}")
```

```
0      0      0      0
1      1      1      1
2      4      8     16
3      9     27     81
4     16     64    256
5     25    125    625
6     36    216   1296
7     49    343   2401
8     64    512   4096
9     81    729   6561
```

Exception

hantera felavbrott

felavbrott

```
>>> fil = open("kudde.txt","r")
Traceback (most recent call last):
File "<pyshell#2>", line 1, in ?
fil = open("kudde.txt","r")
FileNotFoundError:[Errno 2] No such file: 'kudde.txt'
```

Felet heter `FileNotFoundError`

Exception	När uppkommer det?
`FileNotFoundError`	Om man försöker öppna en fil som inte finns.
`SyntaxError`	...skrivit programkod som Python inte kan tolka.
`KeyError`	...försöker använda en nyckel som inte finns i en ordlista.
`NameError`	...använder en variabel utan att ha gett den ett värde.

try...except

Fånga felet genom att införa try...except-sats

```
try:
    fil = open("kudde.txt", "r")
except FileNotFoundError:
    print("Filen finns inte")
```

flera except

```
try:
    tal = int(input("Ge ett heltal: "))
    invers = 1/tal
    print(invers)
except ValueError:
    print("Det där var inte ett heltal!")
except ZeroDivisionError:
    print("Kan inte dela med noll!")
```

använd *aldrig* tom except

```
try:
    tal = int(input("Ge ett heltal: "))
    invers = 1/tal
    print(inver)
except:
    print("Felaktigt tal")
```

fråga

- Hur kan du göra ditt eget program mer användarvänligt genom att lägga till try ... except?

upprepa...

```
def envisInläsning(fråga):
    inläsningOK = False
    while not inläsningOK:
        try:
            tal = int(input(fråga))
        except ValueError:
            print("Inte heltal - försök igen.")
        else:
            inläsningOK = True


envisInläsning("Ge ett heltal: ")
```

...med returvärde

```
def envisInläsning(fråga):
    inläsningOK = False
    while not inläsningOK:
        try:
            tal = int(input(fråga))
        except ValueError:
            print("Inte heltal - försök igen.")
        else:
            inläsningOK = True
    return tal

fråga = "Hur gammal är du? "
tal = envisInläsning(fråga)
print(tal)
```

referenser

att kopiera en lista

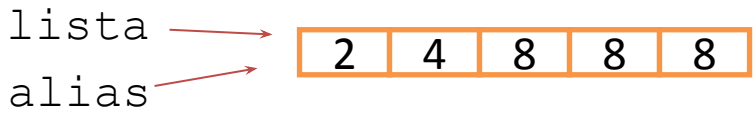
referenser

En listvariabel har en *referens* till listan.

Vid tilldelning är det referensen som kopieras - inte listelementen.

```
alias = lista
```

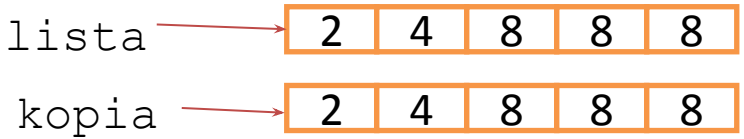
ger följande resultat:



kopiera en lista

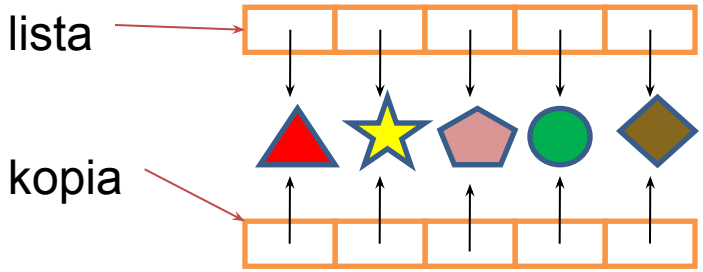
För att få en *kopia* av hela listan kan man använda `copy` i modulen `copy`:

```
from copy import *
kopia = copy(lista)
```



lista av objekt

Om det är objekt i listan kopierar `copy` *referenserna* till varje objekt.

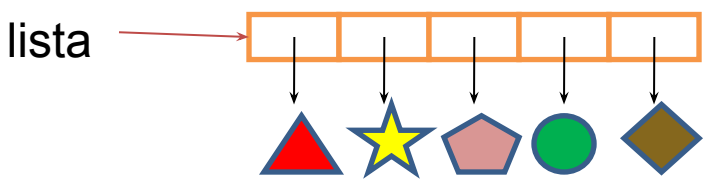


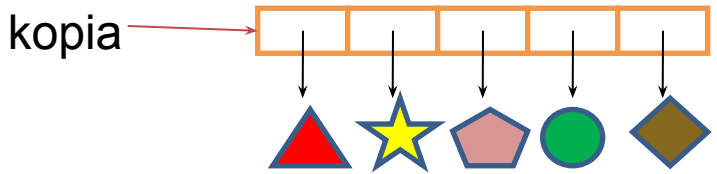
kopiera en lista av objekt

Använd `deepcopy` för att

även göra kopior av *objekten*.

```
from copy import *
kopia = deepcopy(lista)
```





hjälp

med P-uppgiften

tips

Pythontutor: http://pythontutor.com/visualize.html

allmänhandledning

Zoomrum: https://kth-se.zoom.us/j/63569345889

Kö: https://queue.csc.kth.se/Queue/Allmanhandledning