Napake in lov nanje

Težave z deljenjem z nič

- `a = int(input("Kaj delimo: "))`
- `b = int(input("S čim: "))`
- `rez = a / b`
- `print(a, '/', b, '=', rez)`

```
>>>
Kaj delimo: 3
S čim: 4
3 / 4 = 0.75
>>>
```

```
>>>
Kaj delimo: 3
S čim: 0
Traceback (most recent call last):
  File "E:/PrJ/OOP/primerNap.py", line 3, in <module>
    rez = a / b
ZeroDivisionError: int division or modulo by zero
>>>
```

Varovalna mreža

- try ... except

```
a = int(input("Kaj delimo: "))
b = int(input("S čim: "))
try :
    rez = a / b
    print(a, '/', b, '=', rez)
except :
    print('NAPAKA: Deljenje z 0 ni možno')
print('Nadaljevanje programa')
```

```
>>>
Kaj delimo: 3
S čim: 4
3 / 4 = 0.75
Nadaljevanje programa
```

```
>>>
Kaj delimo: 3
S čim: 0
NAPAKA: Deljenje z 0 ni možno
Nadaljevanje programa
>>>
```

Try ... except

- `try :`
  - Poskusi izvesti stavke znotraj tega bloka
  - Če je vse ok, nadaljuj za `try/except`
- `except:`
  - Če pride do napake, nadaljuj tukaj
- Običajno nadaljevanje
- Seveda gre včasih /v zgledu prej/ tudi s stavkom `if`
  - Kako (v prejšnjem zgledu)?

Kaj pa

```
>>>
Kaj delimo: 2
S čim: 2abla
Traceback (most recent call last):
  File "E:/PrJ/OOP/primerNap.py", line 2, in <module>
    b = int(input("S čim: "))
ValueError: invalid literal for int() with base 10: '2abla'
>>>
```

- Napaka pri vnosu! Ni v varovalni mreži!
- Kako bi naredili zadevo s stavkom if?

Povečamo varovalno mrežo

```
try :
    a = int(input("Kaj delimo: "))
    b = int(input("S čim: "))
    rez = a / b
    print(a, '/', b, '=', rez)
except :
    print('NAPAKA: Deljenje z 0 ni možno')
print('Nadaljevanje programa')
```

```
>>>
Kaj delimo: 3
S čim: 4
3 / 4 = 0.75
Nadaljevanje programa
```

```
>>>
Kaj delimo: 3
S čim: 2a
NAPAKA: Deljenje z 0 ni možno
Nadaljevanje programa
>>>
```

Ampak

- Čudno obvestilo
- Saj lahko zadeve razbijemo na več različnih varovalnih mrež
- In v bloku except ni nujno, da le izpisujemo!
- a = ...
- b = ...

Varovalna mreža III

```
try :
    a = int(input("Kaj delimo: "))
except :
    print('To ni število! Vzel bom kar 1')
    a = 1
ok = False
while not ok :
   try :
     b = int(input("S čim: "))
     ok = True
   except :
     print('To ni število! Vnesi ponovno')
# imamo tako a, kot b
try :
    rez = a / b
    print(a, '/', b, '=', rez)
except :
    print('NAPAKA: Deljenje z 0 ni možno')
print('Nadaljevanje programa')
```

```
>>>
Kaj delimo: 3
S čim: a
To ni število! Vnesi ponovno
S čim: 0
NAPAKA: Deljenje z 0 ni možno
Nadaljevanje programa
>>>
```

```
>>>
Kaj delimo: q
To ni število! Vzel bom kar 1
S čim: 12
1 / 12 = 0.083333333333
Nadaljevanje programa
>>>
```

Try ... except

- `try :`
  - Poskusi izvesti stavke znotraj tega bloka
  - Če je vse ok, nadaljuj za `try/except`
- `except:`
  - Če pride do napake, nadaljuj tukaj

Zgled - Inverz

- Sestavi funkcijo `inverz(seznam, i),` ki vrne inverz i-tega elementa seznama
  - `>>> inverz([2, 3, 7], 1)`
  - `0.5`
  - `>>> inverz([2, 4, 7], 2)`
  - `0.25`
  - `def inverz(seznam, ind = 1):`
  - `  return 1 / seznam[ind – 1]`

Kaj gre lahko narobe

- i-ti element je 0
- i-ti element ni število
- Ni i-tega elementa
- Poskusimo napake "poloviti". Če se zgodijo, naj funkcija vrne kar None.
  - `def inverz(seznam, ind = 1):`
  - `  try :`
  - `    return 1 / seznam[ind – 1]`
  - `  except :`
  - `    return None`

Lovimo le sardine, ne pa cipljev

- Lahko lovimo le določene izjeme
- Lahko predvidimo več varovalnih mrež
- `try:`
  - `...`
- `except VrstaNapake_1:`
  - `Kaj ob napaki 1`
- `except VrstaNapake_2 :`
  - `Kaj ob napaki 2`
- …
- `except VrstaNapake_n :`
  - `Kaj ob napaki n`
- In še marsikaj

Zgled z inverzom

- Sestavi funkcijo `inverzV2(seznam, i),` ki vrne inverz i-tega elementa seznama
  - Sedaj pa poskrbimo, da bo uporabnik zvedel, kaj ga je "lomil"
- Omenjene napake so različne vrste!
- Kakšne:
  - Naredimo jih in poglejmo, kaj sporoči Python!