Ugani število

Napišimo program, ki si “izmisli” naključno število, potem pa nas v zanki sprašuje, katero je izmišljeno število. Če število uganemo, nas program pohvali in se zaključi, če pa ga ne uganemo, nam predlaga, da poskusimo še enkrat.

Do sedaj ...

```
import random


def ugani(odKje = 1, doKam = 10): # privzete vrednosti
    stevilo = random.randint(odKje, doKam)
    print("Izmislil sem si število med " +
          str(odKje) + " in " + str(doKam) + ". Poskusi ga uganiti!")
    while True:
        st = int(input("Katero število sem si izmislil? "))
        if stevilo == st:
            print("Čestitam! Pa ti si genij!")
            break # izstop iz zanke, saj je konec!
        else:
            print("Ja ... Še se bo treba matrat ...")
    print('Hvala za igro!')
```

Lahko tudi brez else

```
import random


def ugani(odKje = 1, doKam = 10): # privzete vrednosti
    stevilo = random.randint(odKje, doKam)
    print("Izmislil sem si število med " +
          str(odKje) + " in " + str(doKam) + ". Poskusi ga uganiti!")
    while True:
        st = int(input("Katero število sem si izmislil? "))
        if stevilo == st:
            print("Čestitam! Pa ti si genij!")
            break
        # sem pridemo le, če pogoj pri stavku if ni izpolnjen
        print("Ja ... Še se bo treba matrat ...")
    print('Hvala za igro!')
```

Tudi v zanki for

- Napišimo funkcijo, ki bo vrnila seznam recipročnih vrednosti vseh števil v seznamu. Če je v seznamu podatek, ki ni število, zaključimo s tvorbo seznama in seznam dopolnimo do prave dolžine z vrednostjo 1.
- `>>> sezRecVrednosti([2.5, 4, 1, 5, 'bla', 'ble', 3, 5])`
- `>>>[0.4, 0.25, 1, 0.2, 1, 1, 1, 1]`
- Ideja:
  - Začnemo s praznim seznamom recipročnih vrednosti
  - Z znako `for` gremo preko vseh elementov seznama
  - Če nam uspe izračunati inverz (varovalni blok!), seznamu recipročnih vrednosti dodamo ustrezno vrednost, drugače izstopimo iz zanke
  - Seznamu dodamo ustrezno število 1

Z break in try

```
def sezRecVrednosti(seznam):
  sezRecVr = []
  for x in seznam:
     try:
        sezRecVr = sezRecVr + [1/x]
     except:
        break   # ko naletimo na nepr. podatek, izstopimo iz zanke
  # dopolnimo do ustrezne dolžine
  sezRecVr = sezRecVr +
               [1] * (len(seznam) – len(sezRecVr))
  return sezRecVr
```

Kontrola tipov

- Da preverimo tip podatka
- Metoda `isinstance`
- Vrne `True` ali `False`
- `isinstance(len('bla'), str)  # false`
- `isinstance(x, int) # ali je v x podatek tipa int`
- Če je tip `int` ali `float`, seznamu recipročnih vrednosti dodamo ustrezno vrednost
- `isinstance(podatek, (int, float))`
- **`isinstance`**
- `isinstance(<izraz>, imeTipa)`
- `isinstance(<izraz>, <nabor imen tipov>)`

Z isinstance

```
def sezRecVrednosti(seznam):
  sezRecVr = []
  for x in seznam:
     if not isinstance(x, (int, float)):
       break   # ko naletimo na nepr. podatek, izstopimo iz zanke
     sezRecVr = sezRecVr + [1/x]
  # dopolnimo do ustrezne dolžine
  sezRecVr = sezRecVr +
                [1] * (len(seznam) – len(sezRecVr))
  return sezRecVr
```

Kaj pa 0?

- Pozabili smo na 0
- Kaj se zgodi, če imamo podatek 0?
- Recimo, da moramo 0 v seznamu preskočiti (če je v prvem (dobrem) delu, torej
- `>>> sezRecVrednosti([2.5, 0, 4, 0, 1, 5, 'bla', 'ble', 3, 5])`
- `>>>[0.4, 0.25, 1, 0.2, 1, 1, 1, 1, 1, 1]`

Continue

- Prekine tekoče izvajanje zanke in gre na naslednji korak izvajanja

Z break in continue

```
def sezRecVrednosti(seznam):
  sezRecVr = []
  for x in seznam:
     if not isinstance(x, (int, float)):
       break  # ko naletimo na nepr. podatek, izstopimo iz zanke
     if x == 0 :
       continue # 0 le preskočimo
     sezRecVr = sezRecVr + [1/x]
  # dopolnimo do ustrezne dolžine
  sezRecVr = sezRecVr +
              [1] * (len(seznam) – len(sezRecVr))
  return sezRecVr
```

Igra ugibanja:
omejimo število poskusov

- Omejimo število možnosti – recimo na 5.
- uvesti moramo števec možnosti.
- Kako?
  - Pred vsakim vpisom števila s strani igralca ga opozorimo na število možnosti.
  - Po uganjevanju in preverjanju rezultata (če uganemo – izstop iz zanke) se števec možnosti zmanjša za ena.
  - Če se zgodi, da se števec zmanjša na nič, moramo igralca na to opozoriti ter zaključiti igro (z break).

```
import random

def ugani(odKje = 1, doKam = 10, poskusov = 5):
    stevilo = random.randint(odKje, doKam)
    print("Izmislil sem si število med " + str(odKje) +
              " in " + str(doKam) + ". Poskusi ga uganiti!")
    moznosti = poskusov
    while True:
        print("Imaš še", moznosti, "možnosti")
        st = int(input("Katero število sem si izmislil: "))
        if stevilo == st:
            print "Čestitam! Pa ti si genij!")
            break # prvi izhod
        print("Ja ... Še se bo treba matrat ...")
        moznosti = moznosti - 1
        if moznosti == 0:
            print("Joj joj joj. Izgleda, da ne bo šlo...")
            print("Število je " + str(stevilo))
            break # in še drugi
    print('Hvala za igro!')
```