Signali (u UNIX-u)

O signalima u UNIX-u

- signale primaju procesi
- signale šalje jezgra, izravno (iz svojih razloga) ili kao posrednik
- signale upućuje (izvor signala):
  - jezgra OSa (npr. prilikom greške u pristupu memoriji – `SIGSEGV`)
  - proces sam sebi (npr. nakon isteka vremena `SIGALRM`)
  - drugi proces (preko sučelja OS-a, npr. funkcijom `kill`)
  - korisnik (`Ctrl+C`, preko sučelja OS-a, naredbom `kill` iz ljuske)
- analogno prekidnim signalima na razini procesora (sklopovlja), signali su mehanizam na razini OS-a (primaju ih procesi)
- kao i prekidi i signali služe za obradu asinkronih pojava (proces radi nešto drugo kad mu se signal uputi)
- signali se mogu ignorirati (ne svi) ili su povod akciji
- **proces može reagirati na signal na sljedeće načine:**
  - **prihvatiti i obraditi signal pretpostavljenom funkcijom (npr. prekid izvođenja na SIGINT Ctrl+C)**
  - **prihvatiti i obraditi signal zadanom funkcijom (zadana u programu)**
  - **zadržati signal (za eventualnu kasniju obradu) (NE RADI u Pythonu)**
  - **ignorirati signal**
- izuzetak od gornjeg pravila je signal `SIGKILL (9)` koji uništava proces
- signal ne sadrži nikakve dodatne informacije (osim u nekim proširenjima)
- (NE RADI u Pythonu) pri prihvatu signala ponašanje procesa za taj signal se mijenja u "zadržati signal" dok se on ne obradi; kad obrada završi isti se signal ponovno može prihvatiti na isti način (slično kao kod prihvata prekida: prvo se zabranjuje daljnje prekidanje...)
- UNIX ima oko 30 signala (simbolička imena nalaze se u "signal.h"), neki od njih:
  - `SIGINT` – `Ctrl + C`; `SIGQUIT` – `Ctrl + \ (ž)`; `SIGTSTP` – `Ctrl + Z`
  - `SIGALARM`, `SIGTERM`, `SIGKILL`
- (info) MS Windowsi ne ponašaju se isto za signale (koriste nešto drugo)

Labosi – "Prekidi i signali" u Pythonu

- prekide simuliramo signalima
- ponašanje procesa za signal definiramo funkcijom: `signal.signal()`
- `signal.signal ( sig, obrada )`
  - registrira funkciju `obrada` za signal `sig`
  - tj. kada se idući puta procesu uputi signal `sig` pozvat će se funkcija `obrada`
  - funkcija `obrada` treba biti definirana tako da prima dva parametra: broj signala koji ju poziva te okvir stoga (ovo drugo ne koristimo)
    - `def obrada ( sig, okvir ):`
- `signal.signal ( sig, signal.SIG_IGN )`
  - signal `sig` se ignorira (kao da i nije došao)

Primjer sa signalima u Pythonu

```
import signal, time

def obrada ( sig, frame ):
    print("Pocetak obrade signala")
    for i in range(10):
        print ( "U obradi signala " + str(sig) )
        time.sleep(0.5)
    print("Kraj obrade signala")

def main():
    print("Pocetak programa")

    signal.signal ( signal.SIGINT, obrada )

    for i in range(1,11):
        print("Glavna funkcija: iteracija " + str(i) + " (Ctrl+C za SIGINT)")
        time.sleep(1)

    print ( "Kraj programa" )

if __name__ == "__main__":
    main()
```