Računalniški praktikum I

Standardni vhod, izhod, napaka, preusmeritve in cevovodi

Vida Groznik

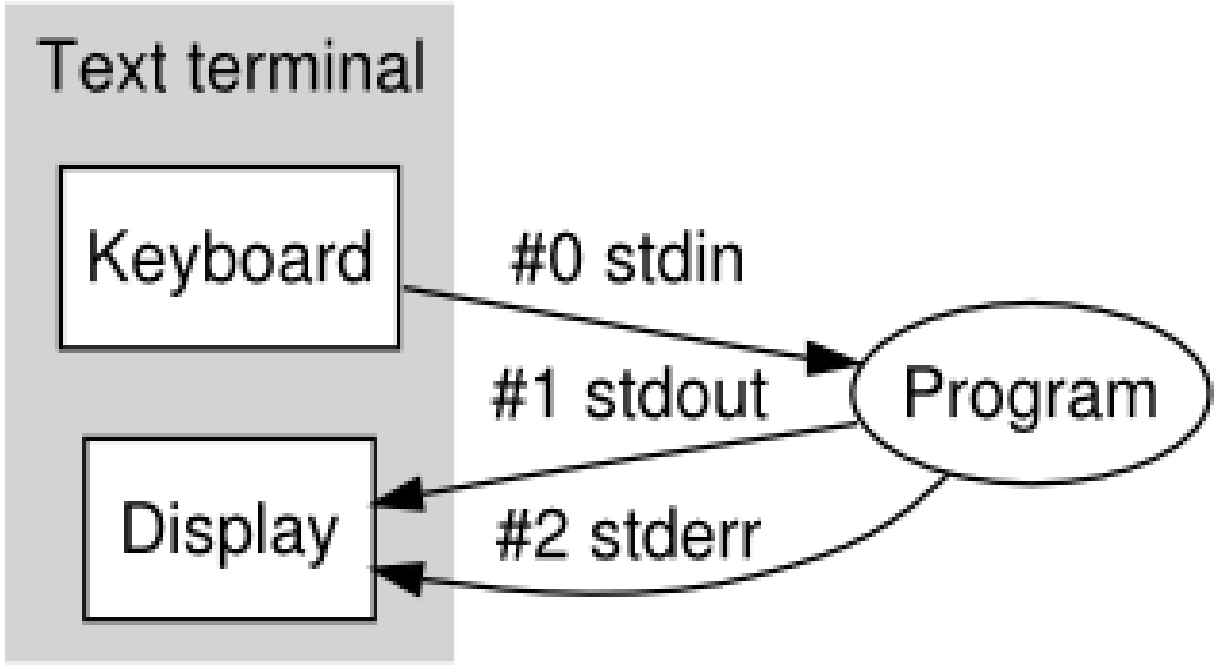
Standardni tokovi

- stdin
  standardni vhod je vhod preko katerega uporabnik lupini sporoča ukaze.
- stdout
  standardni izhod je izhod preko katerega program uporabniku vrača povratno informacijo.
- stderr
  standardna napaka je tok, kamor programi pošiljajo obvestila o napakah izvajanja.

Prikaz delovanja tokov

- Razlika med standardno napako in izhodom:

  napaka ni shranjena v medpomnilnik in se izpiše samo, ko je kaj narobe; vhod in napaka se izpisujeta privzeto na ekran – standardni izhod.



Preusmerjanje tokov

- $ <ukaz> > <datoteka>
  preusmeri izhod v datoteko (ne na ekran), ki jo še ustvari, če ta ne obstaja
- $ <ukaz> < <datoteka>
  prejema podatkov iz datoteke (ne iz tipkovnice)
- $ <ukaz> < <v_datoteka> > <i_datoteka>
  prejema podatke iz datoteke in preusmeri izhod v datoteko, ki jo še ustvari, če ta ne obstaja
- $ <ukaz> 2> <datoteka>
  preusmeri napake v datoteko, ki jo še ustvari, če ta ne obstaja
- $ <ukaz> &> <datoteka> = <ukaz> 1> <datoteka> 2> <datoteka>
  preusmeri napake in izhod v datoteko, ki jo še ustvari, če ta ne obstaja
- $ <ukaz> >> <datoteka>
  preusmeri izhod na konec datoteke, če ta obstaja ali pa jo ustvari, če ta ne obstaja

Cevovod

- Izhod enega programa preusmerimo v vhod drugega programa.
- Označen s pokončnico: |.

$ <ukaz1> [argumenti] | <ukaz2> [argumenti] ...

  - ukaz2 dobi kot vhod tisto, kar bi ukaz1 izpisal drugače na ekran

Ukaz tee

- Ukaz tee nam preusmeri standardni izhod na dva dela:
  - standardni izhod ter
  - datoteko.
- Predstavljamo si ga kot črko T, ki en tok razdeli na dva.

  $ <ukaz> | tee [-a] <datoteka>

  Stikalo -a doda na konec obstoječe datoteke.

Naprava /dev/null

- Mnogokrat se zgodi, da želimo izhod in/ali napako preusmeriti “digitalna nebesa”. Torej se izhod in/ali napaka po sporžitvi ukaza izgubita v črni luknji

  $ <ukaz> > /dev/null
  $ <ukaz> 2> /dev/null
  $ <ukaz> &> /dev/null

Izvrševanje ukazov v zaporedju

- $ <ukaz1> && <ukaz2>
  Če se prvi ukaz izvrši, izvrši še drugega

- $ <ukaz1> || <ukaz2>
  Če se prvi ukaz ne izvrši, izvrši še drugega

- $ <ukaz1> ; <ukaz2>
  Izvrši drugi ukaz, ko se prvi konča (lahko poljubno mnogo ukazov)