Pokretanje programa u OS-u

Ljuska, varijable okoline, pokretanje programa s fork+exec

Informativno! Ne pita se!

(osim na FER-u gdje je to dio lab2)

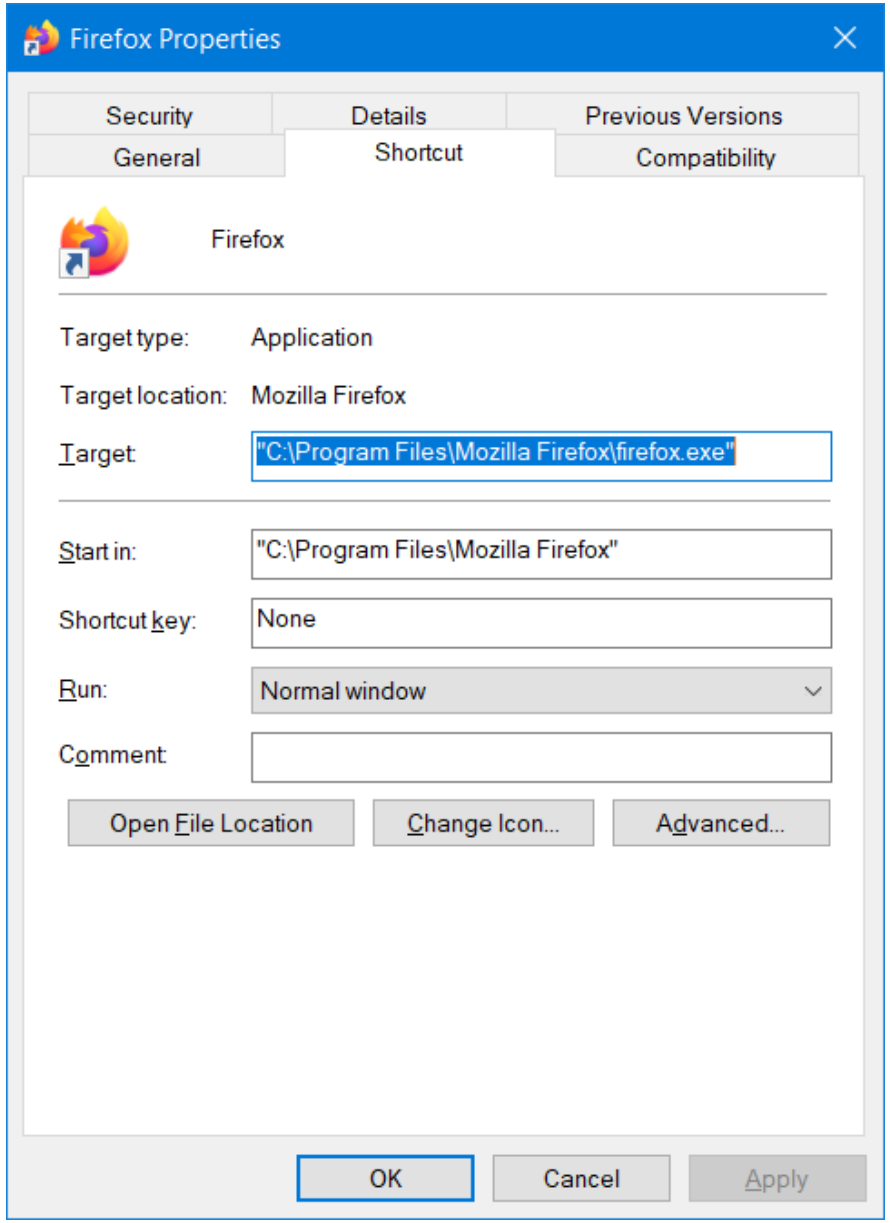
Kako korisnik pokreće programe?

- Koriste se dodatni pomoćni programi
  - programi koju upravljaju grafičkim sučeljem
  - preko naredbene ljuske (Command Prompt, shell)

- Kako ti dodatni programi pokreću tražene programe?
  - koriste sučelje OS-a
    - npr. CreateProcess() na Windows operacijskim sustavima
    - npr. **fork() + execvpe()** na Linuxu

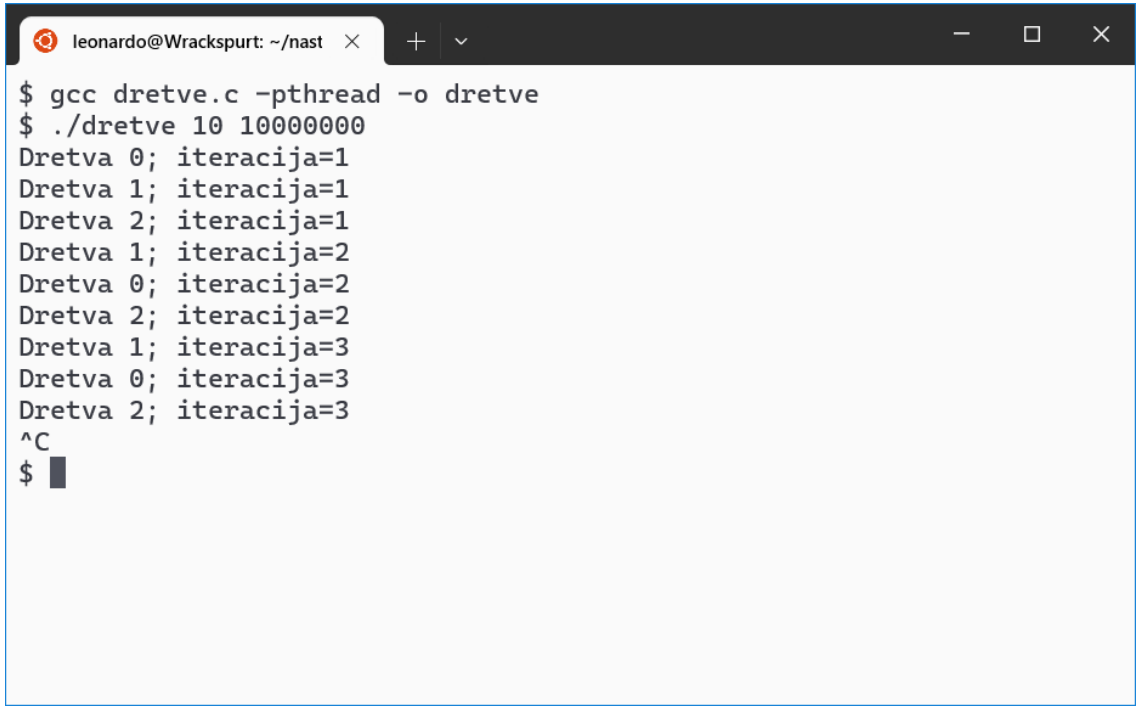
Pokretanje kroz grafičko sučelje

- „Klik” na poveznicu (*shortcut*)
- Što sve piše u poveznici?
  - putanja do programa
  - početni direktorij
- Je li to dovoljno da program radi?
  - može biti, ovisno o programu
  - mogu se dodati argumenti nakon imena programa („Target”)
  - program može pristupiti **varijablama okoline** koje sadrže dodatne informacije o okolini u kojoj se program pokreće



Pokretanje kroz naredbenu ljusku

- Ime programa se ručno upisuje
- Ako nije u trenutnom direktoriju ili „u putanji” onda treba i cijeli put do programa
- Mogu se zadati argumenti nakon imena programa
- Program može pristupiti **varijablama okoline**



- Program može biti s grafičkim sučeljem ili tekstualnim, kada čita i piše u isti terminal ljuske
- Može se pokretati u prvom planu (*foreground*) ili u pozadini (*background*)
- Program se traži u direktorijima koji su navedeni u varijabli okoline *PATH* (ako nije zadana apsolutna/relativna putanja s / ili ./)

Foreground vs Background (UNIX ljuska)

- Program u prvom planu ima „kontrolu” nad ulazom terminala
  - može čitati znakove koje korisnik upisuje (stdin, npr. scanf)
  - može slati signale „izravno” pokrenutom procesu
- Program iz prvog plana se može postaviti u pozadinu (npr. sa *Ctrl+Z* pauzirati te naredom ljuske *bg*)
- Pri pokretanju programa u pozadini dodaje se & na kraju
- Program u pozadini može ispisivati u terminal, ali ne i čitati
- Obično su to programi koji ne traže interakciju s korisnikom
- Standardni ulaz, izlaz i izlaz za greške se mogu preusmjeriti iz, tj. u datoteku (<>) ili u drugi program (|)
  - `prog < ulaz.txt` – stdin=ulaz.txt
  - `prog > izlaz.txt` – stdout=izlaz.txt
  - `prog1 | prog2` – stdout1=stdin2
  - +puno drugih mogućnosti

Varijable okoline

- oblik: **`ime=vrijednost`**
- Varijable okoline su varijable koje sadrže dodatne informacije kojima mogu pristupiti programi
- Npr. ime korisnika, gdje se nalaze neki direktoriji, programi, tip OS-a i slično
- Na Windowsima se mogu vidjeti kroz grafičko sučelje ili naredbom *set* u *Command Prompt-u*
- Na Linuxu s više naredbi, npr. *printenv*

```
C:\Users\Student>set
(samo neke od varijabli)
COMPUTERNAME=RACUNALCE
OS=Windows_NT
Path=C:\Program Files\[+puno toga]
PROCESSOR_ARCHITECTURE=AMD64
SystemDrive=C:
USERNAME=Student
windir=C:\WINDOWS
```

```
$ printenv
(samo neke od varijabli)
NAME=Racunalce
PWD=/home/student
LOGNAME=student
HOME=/home/student
USER=student
PATH=/usr/local/sbin: [+puno toga]
_=/usr/bin/printenv
```

Dohvat/postavljanje varijable okoline u C-u

- U ljusci (*bash*) varijabla se postavlja sa: `export ime-varijable=vrijednost`
- Više načina dohvata, najjednostavniji preko funkcije *getenv()*
- Primjer dohvata:
  - neka postoji varijabla okoline: `USER=student`

    ```
    char *ime = getenv("USER");
    ```
  - `ime` pokazuje na "`student`", vrijednost varijable `USER`
  - ako varijable nema povratna vrijednost je `NULL`
- Postavljanje varijable okoline sa putenv()
  - primjer: `putenv("ULAZ=ulazna-datoteka.txt");`
  - postavlja se samo za trenutni proces, ne za druge postojeće procese!
  - nakon postavljanja, novi procesi stvoreni od ovog nasljeđuju te varijable

Pokretanje programa s fork+exec

- Uobičajeno pokretanje program iz C-a
  1. fork() – stvori novi proces iz postojećeg (dupliciraj)
  2. u novom procesu:
     a) prilagodi sve potrebno (ulaz, izlaz, varijable okoline, ...)
     b) exec* – unutar ovog procesa pokreni program
- Nekoliko inačica exec* funkcija:
  - execl, execlp, execle, execv, execvp, execvpe
  - npr. `int execvpe(char *file, char *argv[], char *envp[]);`
    - `file` – ime programa (npr. `"./dretve"`)
    - `argv` – argumenti (npr. `"./dretve", "10", "1000000", NULL`)
    - `envp` – varijable okoline (npr. `"var1=a", "var2=b"` itd)

Primjer pokretanja programa

```
switch(fork()) {
      case -1: fprintf(stderr, "fork - greska\n");
               exit(-1);
      case 0:  char *args[] = {"./dretve", "10", "1000000", NULL};
               execvp("./dretve", args);
               fprintf(stderr, "execvp – greska!\n");
               exit(-1);
}
```

- Postojeći proces (stvoren sa fork) se „prepiše” novim pokrenutim s exec*
- Nema povratka iz exec* funkcije, osim u slučaju da nije uspješno obavljena