

Prevođenje programskih jezika – beleške sa predavanja Leksička analiza

Milan Banković

*Matematički fakultet,
Univerzitet u Beogradu

Jesenji semestar 2025/26.

Pregled

1 Leksička analiza

Funkcije leksičkog analizatora

Leksički analizator obavlja sledeće zadatke:

- čitanje ulaza
- prepoznavanje leksema i pridruživanje tokena
- uklanjanje komentara

Čitanje ulaza

Čitanje ulaza

- Čitanje ulaza je vremenski najzahtevniji posao
- Mora biti efikasno implementirano
 - ne koristi se baferovanje standardne C biblioteke, jer nije adekvatno za tu svrhu
 - implementira sopstveni bafer koji omogućava vraćanje više od jednog karaktera na ulaz
 - ovaj bafer se popunjava funkcijama za čitanje niskog nivoa (npr. `read()` pod UNIX-om)

Prepoznavanje leksema

Gramzivi algoritam

- Za svaki tip leksema, analizator sadrži po jedan PDKA koji prepoznaje odgovarajući jezik
 - C-implementacija ovih automata se generiše automatski na osnovu regularnih izraza, korišćenjem odgovarajućeg alata, poput [Lex-a](#)
- Prilikom čitanja karaktera sa ulaza, svi automati prolaze kroz odgovarajuća stanja
- U svakom trenutku pamtimo poslednju poziciju na ulazu za koju je neki od automata bio u završnom stanju
- Kada svi automati uđu u stanje greške, vraćamo se na zapamćenu poziciju, a pročitane karaktere nakon te pozicije vraćamo u ulazni bafer (od njih počinje prepoznavanje sledeće lekseme)
- Na zapamćenoj poziciji se završava prepoznata leksema, a automat koji je prepoznao određuje token koji joj pridružujemo
 - U slučaju da je više automata bilo u završnom stanju u tom trenutku, biramo jedan od njih na neki unapred fiksiran način
 - Na primer, u slučaju Lex-a, prioritet imaju oni tokeni čiji su regularni izrazi definisani ranije prilikom opisivanja leksičkog analizatora
 - Na primer, reč **for** će biti prepoznata i kao identifikator i kao ključna reč **for**, ali obično želimo da prednost ima ovaj drugi slučaj
- Pridruženi token se prosleđuje sintaksnom analizatoru, a po potrebi i sama leksema

Prepoznavanje leksema

Posledice gramzivog algoritma

- Gramzivi algoritam uvek prepoznaje najdužu leksemu počev od trenutne pozicije na ulazu koja se uklapa u neki od regularnih izraza koji opisuju klase leksema datog programskog jezika
- Otuda će u slučaju da na ulazu imamo npr. `==`, leksički analizator prepoznati leksemu `==`, a ne dve lekseme `=` jednu za drugom
- Slično, za ulaz `x+++y`, leksički analizator će prepoznati lekseme `x`, `++`, `+` i `y`, tim redom

Uklanjanje komentara

Uklanjanje komentara

- komentari se iz programa uklanjaju na sledeći način:
 - prepoznavanjem početka komentara ulazi se u **režim komentara**
 - u ovom režimu se deaktiviraju svi automati, a ulaz se ignoriše sve dok se ne pročitava oznaka za kraj komentara (ili znak za novi red, u slučaju jednolinijskih komentara)
 - kada se pročita oznaka za kraj komentara, analizator se vraća u **inicijalni režim** u kome se aktiviraju automati i dalje se postupa po gramzivom algoritmu
- u slučaju programskih jezika koji uključuju pretprocesor (C, C++), komentari se obično uklanjaju ranije, u fazi pretprocesiranja