Popravni drugi kolokvij

Interpretacija programa

24. veljače 2017.

Ovo je "*open book*" kolokvij. Dozvoljeno je korištenje bilo kakvih materijala — bilješke s vježbi, Python help, tutoriali, postovi na online forumima,... — **nastalih prije** kolokvija (npr. dozvoljeno je na *StackOverflow*u naći rješenje nekog zadatka, ali nije dozvoljeno tamo postaviti pitanje kako se rješava neki zadatak). Također, nije dozvoljena komunikacija (razgovor, chat, razmjena bilježaka) **među** studentima.

Potrebne / korisne "hint-datoteke" nalaze se u Github repozitoriju, `http://github.com/vedgar/ip` — prethodno skinite repozitorij! Rješenja zadataka pišite **isključivo** u datoteke `Z1P.py` i `Z2P.py`, te na kraju obje datoteke pošaljite mailom na `veky@math.hr`. Odgovore na pitanja koja se nalaze u zadacima pišite u komentare unutar koda.

Svaka datoteka, uključivo sa svim testovima koji su inicijalno u njoj, treba se moći izvršiti u Pythonu bez grešaka. [Smijete pisati i u nekom drugom programskom jeziku, ali u tom slučaju pišete "od nule". Možete koristiti i neke alternativne reprezentacije objekata iz repozitorija (konačni automati, regularni izrazi,...) ako već postoje na internetu. U tom slučaju citirajte link s kojeg ste to skinuli.] Ako imate neki kod koji po Vašem mišljenju pokazuje ideju rješenja, ali iz nekog razloga ne radi, napišite ga u komentar. Korisno je u datoteku uključiti još neke testove pored onih obaveznih.

Prvi zadatak vrijedi 15 bodova, drugi 20 bodova. Ako ima bilo kakvih nejasnoća u vezi teksta zadatka, pitajte me što prije. Maksimalno vrijeme rješavanja je 180 minuta. Rezultati će biti do sat vremena nakon kolokvija. Sretno!

Veky

Prvi zadatak

- [5b] Implementirajte analognu funkciju funkciji `prirodni` iz KA.py, za nedeterminističke konačne automate.
- [5b] Implementirajte funkciju `ε_ciklus`, koja prima nedeterministički konačni automat i njegovo stanje r1, te vraća postoje li njegova stanja r2, ..., rk takva da je r1 povezan s r2, r2 s r3, …, rk s r1, ε-prijelazima. Moguće je i k=1!
- [5b] Implementirajte funkciju `beskonačna_petlja`, koja prima nedeterministički konačni automat i vraća (neku) riječ njegove abecede takvu da se automat može zavrtjeti u beskonačnoj petlji čitajući tu riječ. Ako takva riječ ne postoji, funkcija vraća `None`.

Drugi zadatak

Programski jezik µLogo ima tri ključne riječi: `FORWARD`, `LEFT` i `REPEAT`. Ključna riječ `FORWARD` može se skratiti u `FD`, a `LEFT` u `LT` (ne mijenjajući tip tokena). U µLogu postoje i prirodni brojevi, te uglate zagrade. Oni mogu biti razdvojeni prazninama, koje se zanemaruju. Sve ostalo je leksička greška. [3b] Napišite potrebne tipove tokena.

[5b] Napišite lekser za µLogo. Primjer: `] 57 BLAbla REPEAT FD` treba generirati tokene `ZATV']'`, `BROJ'57'`, `GREŠKA'BLAbla'`, `REPEAT'REPEAT'`, `FORWARD'FD'`, `KRAJ''`.

µLogo ima tri vrste naredbi: `FORWARD` *`x`* za pomicanje kursora *x* piksela u trenutnom smjeru, `LEFT` *`x`* za rotaciju trenutnog smjera *x* stupnjeva pozitivno, te `REPEAT` *`x`* `[` *`naredbe`* `]` za izvršavanje grupe naredbi *x* puta. Argument *x* predstavlja prirodan broj.

[4b] Napišite beskontekstnu gramatiku za µLogo, nad abecedom koju čine tipovi tokena.
[2b] Dokažite da bi ta gramatika bila višeznačna da iz jezika izbacimo uglate zagrade.
[6b] Napišite parser za µLogo. Parser treba vratiti `tip.AST` strukturu (ili listu njih) ako je ulaz validan µLogo program, a prijaviti detaljnu sintaksnu grešku ako nije.

Primjer: `REPEAT 4 [LT 90 FD 150] LT 30 FD 150 LT 120 FD 150` crta kućicu. Parsira se kao lista `[REPEAT(x=4, naredbe=[LEFT(x=90), FORWARD(x=150)]), LEFT(x=30), FORWARD(x=150), LEFT(x=120), FORWARD(x=150)]`.