# Regularni jezici

|  |
| --- |
|  |
| Ostvaruje se na dva načina: vektorski i listom. Vektorski se definira za svako stanje konačnog automata jedan vektor. U vektoru onoliko elemenata kolko je različitih ulaznih znakova. Učinkovito korištenje memorije postize se listom, ali duze traje računanje novog stanja. |
| Uređena petorka DKA=(Q, ∑, δ, q0, F)  Q- konacan skup stanja, ∑- konacan skup ul znakova, δ- funkcija prijelaza Qx∑->Q , q0-pocetno stanje, F- skup prihvatljivih stanja |
| Provjeravanje dva uvjeta. Uvijet podudarnosti: stanja p i q moraju btii oba prihvatljiva ili neprihvatljiva. Uvijet napredovanja: za bilo koji ulazni zank a vrijdi da su stanja δ(p,a) i δ(q,a) istovjetna. |
| Stanje p je nedohvatljivo ako ne postoji niti jedan niz w € ∑\*, za koje vrijedi da je p= δ(q0, w).Njihovim odbacivanjem dobije se istovjetni atuomat sa manjim brojem stanja. |
|  |
| Q'=Q, q0=q0', F'=F U {q0} ako e-okruzenje(q0) sadrzi barem jedno stanje skupa F, inače F'=F δ' (q,a)=δ (q,a), Va∑ i Vq€Q. |
|  |
|  |
| Screenshot_4  Screenshot_5 |
| Definira se kao bTM' (w)=TM (w), w ulazni niz, b izlaz Moorea za prazni niz b= λ(q0) λ' (q,a)= λ (δ(q,a)), za sve q iz Q i za sve a iz ∑ oba automata imaju istu funkciju prijelaza δ |
| Q'=QxΔ gdje je [q,b]€Q', q€Q i b€Δ q0'=[q0, b0], b0 proizvoljni element iz Δ δ'([q,b], a)=[ δ(q,a), λ(q,a)], q€Q i b€Δ, a€∑ λ'([q,b])=b , q€Q i b€Δ |
| Neka je DKA M=(Q, ∑, δ, q0, F) prihvaca L(M). Za komplement jezika L(M)C moguce je izgraditi DKA M'=(Q, ∑, δ, q0, Q\F) koji prihvaca L(M')={w | δ(q0, w)€(Q\F) }={w | δ(q0, w)nije €F}= ∑\*\{w | δ(q0, w)€ F)}= ∑\*\L(M) = L(M)C . |
| Reg jezik jest neprazan ako i samo ako DKA M prihvaca niz z duljine manje od n, tj. |z|<n Utvrdjivanje nepraznosti se prosiruje algor. Nedohvatljivih stanja. Ako je u skupu dohvatljivih barem jedno prihvatljivo stanje, onda je reg jezik neprazan. |
| Broj stanja e-NKA nikad nije veci od 2|r|, gjde je |r| broj znakova u reg izrazu r. E\_NKA ima samo jedno zavrsno stanje f. I vrijedi da je δ(f, a)= prazan skup Skup δ(q,a) sadrzi najvise jedno stanje za ulazni znak a iz skupa ∑. Dok skup δ(q,e) sadrzi najvise dva stanja. |
| 1. Screenshot_6Screenshot_1   16.1.    Screenshot_2  Screenshot_3 |
| Screenshot_7 |
| Screenshot_8  Screenshot_9 |
| ԑ-NKA = ( Q, ∑, ᵟ , q0 , F )  Q –konačan skup stanja ∑ - konačan skup ulaznih znakova ᵟ - funkcija prijelaza Q x ( ∑ U {ԑ} ) --> 2^Q  q0 – početno stanje F podskup Q – skup prihvatljivih stanja Takoder se definira ԑ-OKRUŽENJE: ԑ-OKRUŽENJE(q) = {p| a stanje p jest stanje q ili ԑ-NKA prijelazi iz stanja q  u stanje p primjenom isključivo ԑ-prijelaza |
| Algoritmi koji se koriste u postupku odbacivanja beskorisnih znakova su: algoritam odbacivanja beskorisnih znakova A ZATIM algoritam odbacivanja nedohvatljivih znakova. Primjena algoritama obrnutim redoslijedom ne će nužno rezultirati i odbacivanjem svih beskorisnih znakova... |
| Screenshot_10 |
| Ako nije moguće iz znaka X generirati niz završnih znakova, tj. Ako ne postoji postupak generiranja X =\*=>wx, gdje je wxϵT\* , onda je znak mrtav. Mrtvi znakovi se traže pomoću algoritna traženja živih znakova: 1. U listu živih znakova stave se lijeve strane produkcija, koje na desnoj  strani nemaju nezavršnih znakova 2. Ako su na desnoj strani produkcije isključivo živi znakovi, onda se nezavršni znak lijeve strane produkcije doda u listu živih znakova 3. Ako nije moguće proširiti listu živih znakova, onda se algoritam zaustavlja. Svi znakovi koji nisu u listi živih znakova su mrtvi znakovi. |
| Ovisno o vrsti konačnog automata koji se konstruira, izgradi se odgovarajući program simulator. Simulator konačnog automata jest program koji slijedno čita znakove ulaznog spremnika . Nakon pročitanog znaka, simulator računa prijelaz u novo stanje na temelju podataka iz tablice prijelaza. Stanje nakon posljednje pročitanog znaka odlučuje da li se ulazni niz prihvaća. Želi li se konstruirati konačni automat za neki drugi jezik , potrebno je izgraditi novu tablicu prijelaza. Generator konačnog automata gradi tablicu na temelju zadanih regularnih izraza. Tablica prijelaza se ugradi u simulator.  ( u pdf / word staviti slike sa stranica 50 i 51 !!!) |
| SVOJSTVO ZATVORENOSTI Slijedi izravno iz zatvorenosti s obzirom na uniju i komplement, što se dokazuje DeMorganovim pravilom: L∩N=(( L∩N)c)c=( Lc∩Nc)c Ako su zadani DKA M1 = ( Q1, ∑, ᵟ1 , q1 , F1 ) i DKA M1 = ( Q2, ∑, ᵟ2 , p1 , F2 ) onda se gradi DKA M = ( Q, ∑, ᵟ , q0 , F ) koji prihvaća L(M) = L(M1) ∩ L(M2) gradi na sljedeći način: 1. Q = Q1 x Q2, stanje [q, p] ϵ Q, gdje je qϵQ1 i p ϵQ2 2. Q0 = [q1, p1] 3. F = F1 x F2, stanje [q, p] ϵ F , gdje je qϵF1 i p ϵF2 4. ᵟ ([q, p], a ) = [ ᵟ1(q,a ) , ᵟ2(p,a)] svaki q ϵ Q1 , svaki p ϵ Q2, svaki a ϵ ∑ |
| Screenshot_11 |

# Konteksno neovisni jezici

|  |
| --- |
| U [lingvistici](http://hr.wikipedia.org/wiki/Lingvistika) i [računarstvu](http://hr.wikipedia.org/wiki/Ra%C4%8Dunarstvo), **kontekstno neovisna gramatika (KNG)** je [formalna gramatika](http://hr.wikipedia.org/wiki/Formalna_gramatika) u kojoj je svaka produkcija oblika  V → *w*  gdje je V [nezavršni znak](http://hr.wikipedia.org/wiki/Zavr%C5%A1ni_i_nezavr%C5%A1ni_znakovi) a *w* niz znakova (string) koji se sastoji od završnih i/ili nezavršnih znakova. Termin "kontekstno neovisna" izražava činjenicu da će nezavršni znak V uvijek biti zamijenjen nizom znakova *w* neovisno o kontekstu u kojem se pojavljuje. [Formalni jezik](http://hr.wikipedia.org/wiki/Formalni_jezik) je [kontekstno neovisni jezik](http://hr.wikipedia.org/wiki/Kontekstno_neovisni_jezik) ako postoji kontekstno neovisna gramatika koja ga generira. |
| *Str 77.* |
| *str 82.* |
| 1. *Str 83.* |
|  |
| *106.str, 110. str* |
| *110.str* |
| *112. str* |
| *115.str* |
| *Str 120.* |
| *Str 121.* |
| *Str 122.* |
| *Str 123.* |
|  |
| Screenshot_12 |
| Screenshot_13 |
| 1. Ukratko opisati osnove značajke parsiranja niza tehnikom rekurzivnog spusta (svojstva programskoj jezika, struktura programa, veza programa i gramatike)   Screenshot_14 |
| 1. 3. Opisati postupa konstrukcije gramatike kojom se dokazuje da su konteksno-neovisni jezici zatvoreni s obzirom na proporciju unije   Screenshot_15 |

# Rekurzivno prebrojivi jezici

|  |
| --- |
| Str 127. Screenshot_16 |
|  |
| Screenshot_17 |
| Screenshot_18 |
| Screenshot_19 |
| Screenshot_20 |
| Screenshot_21 |
| Screenshot_22 |
| JESU...dokazi 5. i 8. pitanje |
| Screenshot_23 |
| Konkatenacija je valjda vezivanje pa se misli na nadovezivanje... O.o  Odgovor u 17. u konetekstnoneovisnim jezicima....dokaz za nadovezivanje |
| Screenshot_24 |
| 1. Screenshot_26Screenshot_25 |

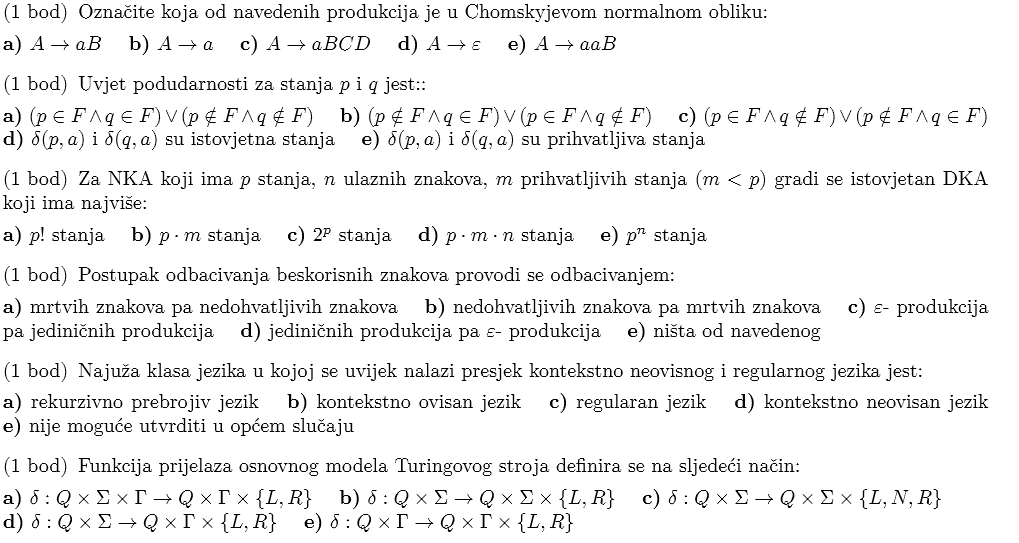
# Konteksno ovisni jezici

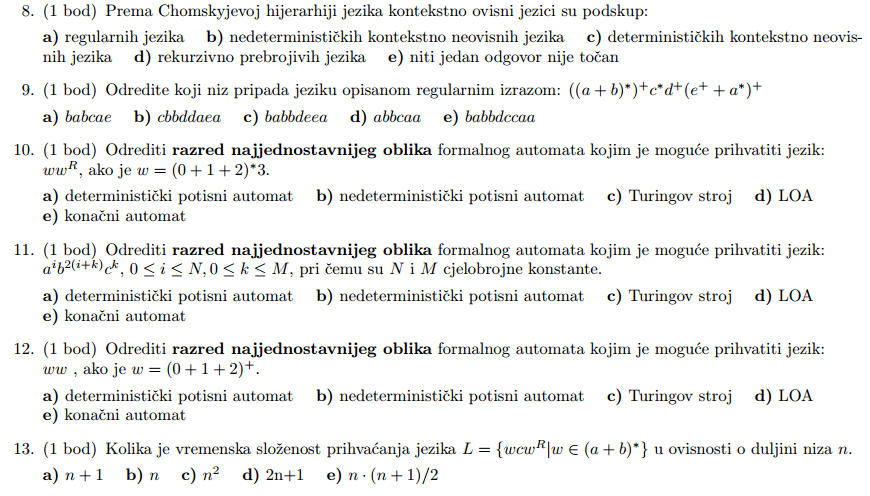
|  |
| --- |
| *168. str* |
|  |
| Str 170. |

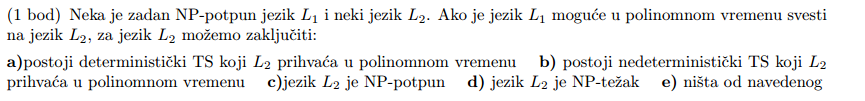
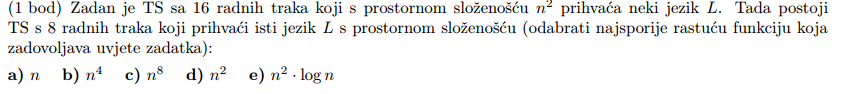
# Razredba jezika, automata i gramatika

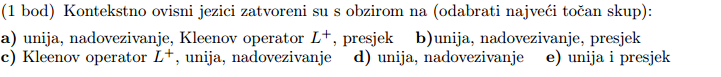
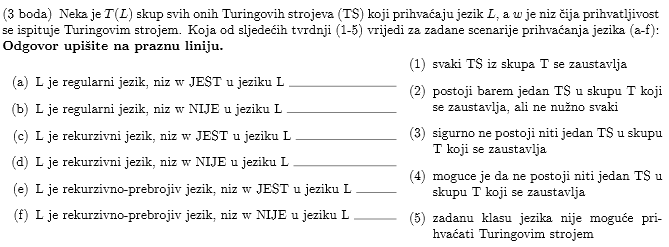
|  |
| --- |
|  |
| REGULARAN JEZIK  1. L(M1) = {} nad abecedom ∑ = {0,1} prihvaća DKA M1: M1 = ( {q} , {0,1} , {δ(q,0) = q , δ(q,1 ) =q} , q , {})  2. L(M2) = {ԑ} nad abecedom ∑ = {0,1} prihvaća DKA M2: M2 = ( {q,p} , {0,1} , {δ(q,0) = p , δ(q,1 ) =p , δ(p,0) = p , δ(p,1 ) =p}, q , {q})  3. L(M3) = ∑\* nad abecedom ∑ = {0,1} prihvaća DKA M3: M3 = ( {q} , {0,1} , {δ(q,0) = q , δ(q,1 ) =q} , q , {q})   DETERMINISTIČKI KONTEKSTNO-NEOVISNI JEZIK   - valjda oni koje prihvaća potisni automat, kaže wikipedija, ako je tako onda Moze i anbn ( n je exponent )  L(M) = {(na)m | n ≥ 1, m ≥ 0 , n ≤ m} n,m eksponenti  Provjeriti i N (M) = {w2wR | w npr. Niz 0 i 1 , tj ( 0 + 1 )\* ,a wR obrnuto, dvojka omogućuje donošenje odluke}  NEDETERMINISTIČKI KONTEKSTNO-NEOVISNI JEZIK  N ( M ) = { wwR | w npr. Niz 0 i 1 , tj ( 0 + 1 )\* ,a wR obrnuto } Općenito palindromi  REKURZIVNI JEZIK  Jezik kojega prepoznaje TS  {ww | w ϵ ∑\* } { wcy |w ϵ∑\* , w različito y }  REKURZIVNO PREBROJIV JEZIK  Jezik koji TS prihvaća,  - Lu -> Univerzalan - L(M) = { a^i b^j c^k | i≠j ili j≠k } - L(M) = { 0^n 1^n | n>0 } - L(M) = { wcy | w,yeΣ\*, w≠y }  NEIZRAČUNLJIV   L = {ai : ai ∉ Mi } , jezik za koji ne postoji TS koji ga prihvaća, ako netko može pronaći konkretan primjer nek javi , i je exponent  - Ld -> Dijagonalan |
|  |
| ***Str 182.*** |
|  |
| ***Str 183.*** |
| ***Str 184.*** |
| **Str 184. i 185.**  **Screenshot_27**  **Screenshot_28** |
| ***Str 184.*** |
| ***Str 187*.** |
| ***Str 194*.** |
| **Str 197.** |
| ***Str 197.*** |

# Zadaci na zaokruživanje







. ****