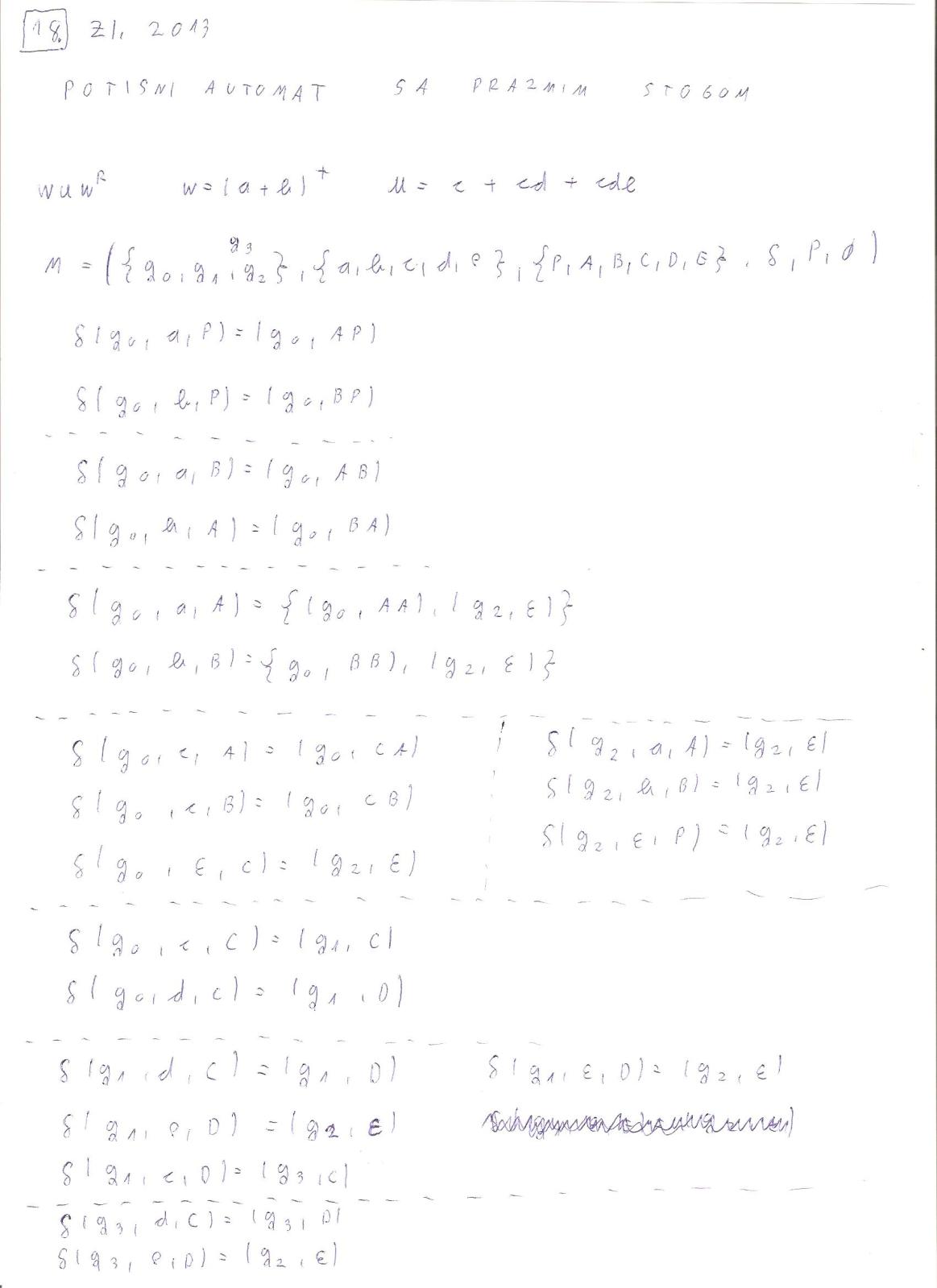
16) nedohvatljiva su p6, p7 nema istovjenih

17) tablica stanja [p0,x], [p1,x], [p2,x], [p0,y], [p1,y], [p1,z] izlazi x,x,x,y,y,z

18)



**19) for(int w=0,w<n;++w)x;**U zadatku bi trebalo biti navedeno koji tip gramatike se trazi, a u suprotnom se vjerojatno trazi ili kontekstno neovisna gramatika ili gramatika neogranicenih produkcija, ovisno o jeziku koji treba generirati. Kako je u ovom slucaju zadani jezik kontekstno ovisan, treba konstruirati gramatiku neogranicenih produkcija (ili kontekstno ovisnu gramatiku, ali mislim da je to bolje izbjegavati). Naime, kljucna stvar u ovom jeziku je da su ova tri niza w jednaka u svakom nizu i to je ono sto jezik cini kontekstno ovisnim, tj. zbog toga jezik nije kontekstno neovisan.  
Pristup konstrukciji gramatika neogranicenih produkcija je dosta slican pristupu konstrukciji Turingovog stroja, samo sto umjesto glave za citanje i pisanje i stanja TS koristimo nezavrsne znakove koji obavljaju vise manje istu funkciju. Ideja je da svaki puta kada generiramo neki znak u prvoj instanci niza w, generiramo neki nezavrsni znak koji ce osigurati da se taj isti zavrsni znak pojavi i u preostalim instancama niza w. Dakle, produkcije bi bile otprilike ovakve (ovo sam sad na brzinu napisao pa ne garamtiram da je 100% tocno, ali zelim samo pokazati ideju).  
  
S -> for(int P=0;<n;++) X;

X -> aY | bY  
Y -> aY | bY | eps  
  
(X i Y generiraju ovaj niz x tj. "naredbu" u petlji pri cemu X osigurava da taj niz nije prazan)

P -> QaA | QbB  
Q -> QaA | QbB | eps  
  
(P osigurava da w nije prazan niz i predaje dalje generiranje znaku Q koji je osim toga identican P... niz w se generira s desna na lijevo tj. uvijek se generira krajnje lijevi znak)

Aa -> aA  
Ab -> bA  
  
(A "putuje" u desno po nizu w)

A=0; -> =0;aA  
A<n;++ -> <n;++a  
  
(A generira znakove "a" u drugoj i trecoj instanci niza w).  
Analogne produkcije naprave se za znak B pa necu to sad pisati.

20) |w2| = |w1| +2|w2|   
**c** 0^i **#** 0^i+2j **#** 0^j **$**odi do lijevog granicnika (c)   
umjesto **0** stavi **N** a nulu prebci desno preko prvog # i moras naletiti na **0** i zapisi kao **I**  
ako naletis na **#** kraj, tako prebacuj sve nule i kad vise nema nule znaci doso si do **#**  
  
odi do desnog granicnika ($)  
umjesto 0 stavi N a nulu prebaci lijevo preko drugog # i moras naletiti na **0** i zapisi **J**ako naletis na **N** ili **#** kraj,   
  
ako si sve prebacio sad jos sve **nule** koje su ostale izmedju **I** i **J**c N^i # I^i **0^j** J^j # J^j $   
treba prebacit desno na J-tove preko # ako si sve rijesio i naletis na prvi # super   
ako nulu trebas prebaciti na $ kraj