1. Koraci postupka analize izvornog programa redom su :

**Leksička, sintaksna i semantička**

1. Za slijedeću kontektsno neovisnu gramatiku S -> aABc; S -> cBAb; A -> aA; A -> ϵ; B -> bB; B -> c izračunajte vrijednost relacije ReduciranZnakom za završni znak c:

**ReduciranZnakom(c,a),ReduciranZnakom(c,b),ReduciranZnakom(c,c),ReduciranZnakom(c,Ʇ)**

1. U LL(1) gramatici, za praznu produkciju A -> α, relacija PRIMJENI izračunava se kao:

**ZAPOČINJE(α)∩SLIJEDI(A)**

1. Za sljedeću gramatiku odredi vrstu: S->Bb, S->cBb, B-> ϵ, B->ab:

**LR(1)**

1. Koraci postupka analize izvornog programa redom su :

**Leksička, sintaksna i semantička**

1. Za slijedeću kontektsno neovisnu gramatiku S -> aABc; S -> cBAb; A -> aA; A -> ϵ; B -> bB; B -> c izračunajte vrijednost relacije ReduciranZnakom za završni znak c:

**ReduciranZnakom(c,a),ReduciranZnakom(c,b),ReduciranZnakom(c,c),ReduciranZnakom(c,Ʇ)**

1. Koliko će produkcija imati zadana gramatika A -> aBa; A -> acBc; B->Ba; B->aA; B->ϵ nakon izbacivanja praznih znakova:

**7**

1. Pri izgradnji potisnog automata za atributnu prijevodnu gramatiku, temeljem produkcije A -> ξbфα, pri čemu su ξ i ф izlazni znakovi, gradi se akcija:

**Izlaz(ξф); Zamijeni(); Pomakni;**

1. Tijekom analize strukture pri analizi tijeka izvođenja programa, analiza strukture obilazi graf tijeka izvođenja programa, traži podgrafove uzorke, zamjeni ih jednim zamjenskim čvorom i gradi:

**Upravljačko stablo**

1. Pri optimiranju procedure međukoda srednje razine koristi se pretvorba rekurzivnih procedura u:

**Petlje**

1. Generiranje ciljnog programa na temelju post fiksnog sustava oznaka: Ako se u međukodu pročita operator, onda generator primjeni akciju:

**Uzmi sa vrha stoga zadani broj operanda , generiraj naredbe ciljnog programa i stavi rezultirajući operand na vrh stoga**

1. Zadana je L-atributno prijevodna gramatika(indeksi su u zagradama): X(n1,i2,i3) -> Y(n4)Z(i5)V(i6,n7,n8)W(n9) gdje su i2,i3,i5 i i6 izvedena svojstva, a n1,n4,n7,n8 i n9 nasljedna. Nasljedno svojstvo n7 moze se racunati na temelju svojstva:

**n1,n4,i5**

1. Zadana je Q-gramatika s produkcijama: S -> bAS; S -> a; A -> ϵ. Odredite skup PRIMJENI(A->ϵ).

**{a,b}**

1. Dio izvornog programa u kojem je važeća deklaracija naziva se \_\_\_\_\_\_\_ deklaracije.

**Djelokrug**

1. Ako je stanje LR(1) parsera označeno stavkama X -> α·{a,b,c} i X -> α·gβ{d,e,f}, parser će učitavanjem znaka ***b*** iz ulaznog niza izvesti akcije:

**Reduciraj**

1. Ako je stanje LR(1) parsera označeno stavkama X -> α·{a,b,c} i X -> α·gβ{d,e,f}, parser će učitavanjem znaka ***e*** iz ulaznog niza izvesti akcije:

**Pomakni**

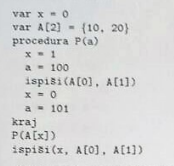
1. Izvođenje naredbi za koje vrijedi da se vrijednost relacije \_\_\_\_\_\_\_ ne mijenja određuje životni vijek pridruživanja imena.

**Okoline**

1. Za sljedeću kontekstno neovisnu gramatiku S -> aABc ; S->cBAb ; A->aA ;A->ϵ; B->bB; B->c; izračunajte vrijednosti relacija *ReduciranZnakom* za završni znak *c*.

**R(c,a), R(c,b), R(c,c), R(c, Ʇ)**

**Tekst se odnosi na zadatke 20.,21.,22.,23. Za programski odsječak (priložena slika) odredite ispis ako se kod poziva potprograma koristi:**

1.  (i) razmjena vrijednosti

**10,20; 0,10,20**

1. (ii) razmjena adrese

**100,20; 0,101,20**

1. (i) razmjena imena

**100,20; 0,101,20**

1. (i) povratna razmjena vrijednosti

**10,20; 0,101,20**