

Prevodioci i interpretatori - Februar 2004.

teorijski deo ispita

1. (a) Kontekstno slobodnom gramatikom opisati deklaraciju slogovnog (**record**) tipa u Paskalu, a zatim je transformisati u ekvivalentnu LL(1)-gramatiku. Slog može sadržati deklaracije prostih i nizovskih promenljivih, kao i pokazivačke promenljive. Deklaraciji može (opciono) prethoditi ime tipa. Na primer,

```
primer = record
    x, y : integer;
    z : array[1..10] of array[1..5] of integer;
    a, b : ^char;
    x1, y2 : real
end;
```

- (b) Konstruisati odgovarajući potisni automat.

Prevodioci i interpretatori - Februar 2004.

teorijski deo ispita

1. (a) Kontekstno slobodnom gramatikom opisati deklaraciju slogovnog (**record**) tipa u Paskalu, a zatim je transformisati u ekvivalentnu LL(1)-gramatiku. Slog može sadržati deklaracije prostih i nizovskih promenljivih, kao i pokazivačke promenljive. Deklaraciji može (opciono) prethoditi ime tipa. Na primer,

```
primer = record
    x, y : integer;
    z : array[1..10] of array[1..5] of integer;
    a, b : ^char;
    x1, y2 : real
end;
```

- (b) Konstruisati odgovarajući potisni automat.

Prevodioci i interpretatori - Februar 2004.

teorijski deo ispita

1. (a) Kontekstno slobodnom gramatikom opisati deklaraciju slogovnog (**record**) tipa u Paskalu, a zatim je transformisati u ekvivalentnu LL(1)-gramatiku. Slog može sadržati deklaracije prostih i nizovskih promenljivih, kao i pokazivačke promenljive. Deklaraciji može (opciono) prethoditi ime tipa. Na primer,

```
primer = record
    x, y : integer;
    z : array[1..10] of array[1..5] of integer;
    a, b : ^char;
    x1, y2 : real
end;
```

- (b) Konstruisati odgovarajući potisni automat.

Prevodioci i interpretatori - Februar 2004.

teorijski deo ispita

1. (a) Kontekstno slobodnom gramatikom opisati deklaraciju slogovnog (**record**) tipa u Paskalu, a zatim je transformisati u ekvivalentnu LL(1)-gramatiku. Slog može sadržati deklaracije prostih i nizovskih promenljivih, kao i pokazivačke promenljive. Deklaraciji može (opciono) prethoditi ime tipa. Na primer,

```
primer = record
    x, y : integer;
    z : array[1..10] of array[1..5] of integer;
    a, b : ^char;
    x1, y2 : real
end;
```

- (b) Konstruisati odgovarajući potisni automat.

- 2 (a) Dokazati da je data gramatika višeznačna (ambiguous):

naredba	→	<i>if</i> uslov <i>then</i> naredba
		<i>if</i> uslov <i>then</i> naredba <i>else</i> naredba
		<i>prom=broj</i>
uslov	→	<i>broj</i> rel <i>broj</i>
rel	→	< > ≤ ≥ =

(b) Konstruisati bar jednu nevišeznačnu gramatiku ekvivalentnu datoj.

- 3 Konstruisati minimalni deterministički automat konačan automat koji prepoznaje reči nad $\Sigma = \{a, b\}$ u kojima se poslednje slovo javlja bar još dva puta prethodno u reči.

- 2 (a) Dokazati da je data gramatika višeznačna (ambiguous):

naredba	→	<i>if</i> uslov <i>then</i> naredba
		<i>if</i> uslov <i>then</i> naredba <i>else</i> naredba
		<i>prom=broj</i>
uslov	→	<i>broj</i> rel <i>broj</i>
rel	→	< > ≤ ≥ =

(b) Konstruisati bar jednu nevišeznačnu gramatiku ekvivalentnu datoj.

- 3 Konstruisati minimalni deterministički automat konačan automat koji prepoznaje reči nad $\Sigma = \{a, b\}$ u kojima se poslednje slovo javlja bar još dva puta prethodno u reči.

- 2 (a) Dokazati da je data gramatika višeznačna (ambiguous):

naredba	→	<i>if</i> uslov <i>then</i> naredba
		<i>if</i> uslov <i>then</i> naredba <i>else</i> naredba
		<i>prom=broj</i>
uslov	→	<i>broj</i> rel <i>broj</i>
rel	→	< > ≤ ≥ =

(b) Konstruisati bar jednu nevišeznačnu gramatiku ekvivalentnu datoj.

- 3 Konstruisati minimalni deterministički automat konačan automat koji prepoznaje reči nad $\Sigma = \{a, b\}$ u kojima se poslednje slovo javlja bar još dva puta prethodno u reči.

- 2 (a) Dokazati da je data gramatika višeznačna (ambiguous):

naredba	→	<i>if</i> uslov <i>then</i> naredba
		<i>if</i> uslov <i>then</i> naredba <i>else</i> naredba
		<i>prom=broj</i>
uslov	→	<i>broj</i> rel <i>broj</i>
rel	→	< > ≤ ≥ =

(b) Konstruisati bar jednu nevišeznačnu gramatiku ekvivalentnu datoj.

- 3 Konstruisati minimalni deterministički automat konačan automat koji prepoznaje reči nad $\Sigma = \{a, b\}$ u kojima se poslednje slovo javlja bar još dva puta prethodno u reči.