Elektrotehnički fakultet u Beogradu

Predmet:	Programski prevodioci 1
Nastavnik:	dr Dragan Bojić, red. prof.
Asistenti:	doc. dr Maja Vukasović mast.inž. Mihajlo Ogrizović mast.inž. Luka Hrvačević
Ispitni rok:	Januar 2026.
Datum:	19.02.2026.

Potpis dežurnog

Kandidat: _____

Broj Indeksa: _____ *Smer:* _____ *Sala:* _____

Ispit traje 150 minuta.
Nije dozvoljeno korišćenje literature.
Prvih sat vremena nije dozvoljeno napuštati ispit.

Zadatak 1	_____/10	*Zadatak 4*	_____/10
Zadatak 2	_____/10	*Zadatak 5*	_____/10
Zadatak 3	_____/10	*Zadatak 6*	_____/10

Ispit: _____/60 **Ukupno:** _____/100

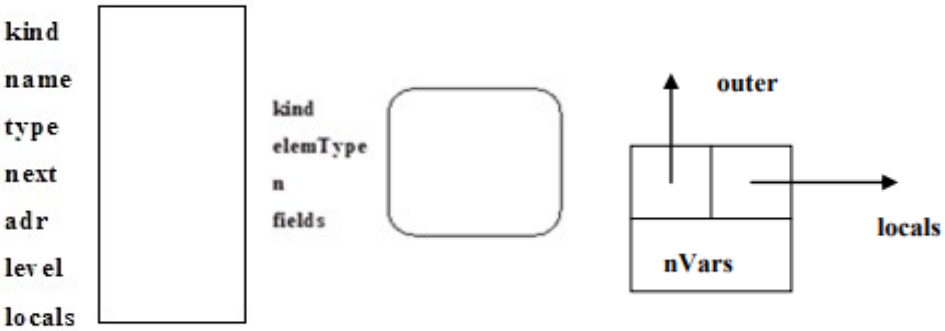
Projekat: _____/40 **Ocena:** _____

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Na pitanja odgovarati **čitko i precizno**. Srećno!

Podsetnik za neke instrukcije Mikrojava bajtkoda

getstatic	s	... → ..., global[s]
putstatic	s	..., val → ...
new	s	... → ..., adr
getfield	s	..., adr → ..., adr.fields[s]
putfield	s	..., adr, val → ...
const	w	... → ..., w
load	b	... → ..., local[b]
store	b	..., val → ...
new	s	... ..., adr
newarray	b	..., n ..., adr
aload		..., adr, index ..., val
astore		..., adr, index, val ...
baload		..., adr, index ..., val
bastore		..., adr, index, val ...
enter b1, b2		
dup		..., val ..., val, val
dup2		..., v1, v2 ..., v1, v2, v1, v2
dup_x1	..,val2, val1	...,val1, val2, val1
dup_x2	val1, val2, val3	...,val3, val1, val2, val3 ...

Podsetnik strukture čvorova tabele simbola.



1) **(10 poena)**
Napisati odgovarajući troadresni međukod u SSA formi i nacrtati graf toka kontrole na nivou osnovnih blokova za dati programski fragment.

```
int a = 0, b = 0, c = 20;
while (c > (a + b) ) {
     if(a % 2 == 0){
          a++;
          continue;
     }
     b++;
     c = c - b;
}
return c;
```

Rešenje:

2) **(10 poena)**
Za regularni izraz `b(a|cb)*c+` odrediti minimalni deterministički automat metodom pozicija.

Rešenje:

3) **(10 poena)**

Zadatak

Jedan AI chatbot dao je sledeći java program pri upitu da napravi jednostavan SHIFT-REDUCE parser za datu gramatiku (deo programa nedostaje na mestu označenom sa ***):

1. S → int IDLIST ;
2. IDLIST → IDLIST , id
3. IDLIST → id

a) [2] Kojoj klasi SHIFT-REDUCE parsera je dati kod najsličniji i zašto?
b) [2] U čemu je glavna razlika između "pravih" SHIFT-REDUCE parsera i ovog koda?
c) [3] Šta treba dodati kodu na mestu označenom sa *** da bi bio kompletan (ostatak koda ne prepisivati)?
d) [3] Pod pretpostavkom da gramatici dodamo smenu S → id, šta bi trebalo izmeniti/dodati u datom kodu da bi se ova nova gramatika ispravno prepoznavala "u duhu" SLR(1) parsiranja (ne pisati po postavci, nego u okviru rešenja napisati izmene koda).

```
import java.util.*;

public class SRParser {
    public static void main(String[] args) {
        List<String> input = new ArrayList<>(Arrays.asList("int", "id", ",", "id", ";"));
        Stack<String> stack = new Stack<>();
        System.out.printf("%-25s | %-25s | %s\n", "Stack", "Input", "Action");
        System.out.println("-----------------------------------------------------------");

        while (true) {
            String action = "";
            int size = stack.size();
            // Pravilo 1: IDLIST "," "id" -> IDLIST
            if (size >= 3 && stack.get(size-3).equals("IDLIST") &&
                stack.get(size-2).equals(",") && stack.get(size-1).equals("id")) {
                stack.pop(); stack.pop(); stack.pop();
                stack.push("IDLIST");
                action = "Reduce (IDLIST, id -> IDLIST)";
            }
            // Pravilo 2: "id" -> IDLIST
            else if (size >= 1 && stack.peek().equals("id")) {
                stack.pop();
                stack.push("IDLIST");
                action = "Reduce (id -> IDLIST)";
            }
            // Pravilo 3: "int" IDLIST ";" -> S
            else if (size >= 3 && stack.get(size-3).equals("int") &&
                stack.get(size-2).equals("IDLIST") && stack.get(size-1).equals(";")) {
                stack.pop(); stack.pop(); stack.pop();
                stack.push("S");
                action = "Reduce (int IDLIST ; -> S)";
            }
            // ***
            } else {
                break; // Nema više mogućih poteza
            }
            System.out.printf("%-25s | %-25s | %s\n", stack, input, action);
        }
        if (stack.size() == 1 && stack.peek().equals("S")) {
            System.out.println("Ishod: USPEŠNO");
        } else {
            System.out.println("Ishod: GREŠKA");
        }
    }
}
```

Rešenje:

4) **(10 poena)**

Za sve sekvence oblika $a^{3k}\ (bc)^k d^k$, $k \geqslant 0$ potrebno je:

a) konstruisati gramatiku koja opisuje ovakve sekvence

b) konstruisati potisni automat koji opisuje ovakve sekvence

Rešenje:

5) **(10 poena)**
Data je sledeća gramatika:

```
<S> → <S> y
<S> → <A>
<S> → ε
<A> → p <A> q
<A> → p <A> r <A>
<A> → ε
```

a) [5] Nalaženjem SELECT skupova proveriti da li je data gramatika tipa LL(1). Ako nije transformisati je u LL(1).
b) [5] Korišćenjem metode rekurzivnog spusta realizovati parser nad gramatikom dobijenom pod a).

Rešenje:

6) **(10 poena)**

Za program Feb2026 prikazati izgled tabele simbola u tačkama T1 i T2. Nije potrebno crtati universe opseg.

Rešenje:

```
program Feb2026
      const int pet = 5;
      int[] arr;
      class A {
            int b;
      {
            int met1(int a) {
                  return a+b;
            }
            int met2(int m) {
                  return m+b;
            }

      } }
      class B extends A {
            int c;
      {  *T1*
            int met1(int d) {
                  return b+c+d;
            }
         *T2*
      } }
{
      void main()
            A a;
      {
            arr = new int[2];
            arr[0] = jedan;
            a = new B;
            a.b = arr[0];
      }
}
```

Dodatni prostor za rad: