

Precizări:

- Fiecare subiect valorează câte 10p. **Punctajul NU se acordă în absența justificării răspunsului!**
- Este suficientă rezolvarea a 10 itemi dintre cei 12 pentru nota maximă; se acordă și punctaj parțial.
- Punctajul suplimentar reprezintă **bonus**, ce poate compensa punctajul de pe parcurs.

1. Fie specificația lexicală de mai jos:

$$((10|011)^+01)^*1 \quad (1)$$

$$(01(10|011)^+)^*1 \quad (2)$$

$$10011(01)^* \quad (3)$$

(a) Care este secvența de reguli lexicale (1–3) utilizate în analiza următorului șir?

01101001011011011001111.

(b) Dați un exemplu de șir care **nu** poate fi analizat de această specificație.

2. Rescrieți gramatica de mai jos într-o variantă factorizată la stânga și fără recursivitate la stânga.

$$E \rightarrow \text{Dispatch} \mid E + E \mid id$$

$$\text{Dispatch} \rightarrow E.id() \mid id()$$

Subiectele 3–4 vizează gramatica de mai jos:

$$S \rightarrow aBAb \mid Bc$$

$$A \rightarrow Bd \mid e$$

$$B \rightarrow fS \mid \varepsilon$$

3. Determinați mulțimile *First* și *Follow*, construiți tabelul de analiză *LL(1)* și conchideți dacă gramatica este într-adevăr *LL(1)*.

4. (a) Construiți automatul de analiză *LR* doar până întâlniți o stare cu un conflict *LR(0)*, și precizați tipul conflictului și itemii implicați.

(b) Precizați dacă și de ce dispăre conflictul depistat mai sus, asumând euristica *SLR(1)*.

Subiectele 5–9 vizează secvența Cool de pe verso.

5. Care este conținutul contextelor de tipare pentru obiecte, *O*, în punctul *P1*, respectiv pentru metode, *M*?

6. (a) Aplicați regula potrivită pentru verificarea tipului expresiei `new SizedSet.init(5)`, din metoda `main` a clasei `Main`. Utilizați manualul Cool.

(b) Păstrând restul codului neschimbat, este posibilă modificarea tipului întors de metoda `remove` de la `SELF_TYPE` la `Set`?

7. (a) Descrieți reprezentarea în memorie a obiectelor prototip pentru clasele `Set` și `SizedSet`, incluzând tabelele de metode.

(b) Presupunând că toate informațiile necesare evaluării corpurilor metodelor se depun pe stivă, care este dimensiunea minimă a înregistrării de activare pentru metoda `process` din clasa `Set`?

8. Pentru expresia `str.length() <= n`, din metoda `process` a clasei `Set`, este propusă următoarea secvență de cod MIPS, unde registrul `$s0` conține adresa lui `self`, iar convențiile de organizare a înregistrării de activare sunt cele din enunțul temei 3. Completați cele 4 spații libere cu valorile corecte și justificați!

```
1      lw      $a0 __($s0)
2      <verificare dispatch on void>
3      lw      $t1 8($a0)
4      lw      $t1 __($t1)
5      jalr     $t1
6      sw      $a0 0($sp)
7      addiu    $sp $sp -4
8      lw      $a0 __($fp)
9      lw      $t1 4($sp)
10     addiu    $sp $sp 4
11     lw      $t1 12($t1)
12     lw      $t2 12($a0)
13     la       $a0 bool_const1
14     __       $t1 $t2 L1
15     la       $a0 bool_const0
16 L1:
```

9. (a) Pentru punctul *P2* din execuția programului, realizați o diagramă a obiectelor alocate pe *heap*, cu evidențierea referințelor, presupunând că niciun obiect nu a fost încă eliberat. Care obiecte sunt *reachable*?

(b) Extindeți răspunsul de mai sus pentru punctul *P3*.

(c) Cum arată *heap*-ul după aplicarea strategiei *stop-and-copy* în punctul *P3*?

10. Descrieți, pas cu pas, aplicarea strategiei *local value numbering* asupra *basic block*-ului de mai jos. Completați tabelul valorilor și indicați cum se modifică fiecare instrucțiune, unde este cazul.

```
1  x := 1
2  y := x + 2
3  z := a + b
4  u := 3 * z
5  v := z * y
6  w := u - v
```

Subiectele 11–12 vizează codul intermediar de mai jos:

```
1      u := 1
2      v := 2
3      if w < 2 goto L2
4 L1:
5      a := v * 2
6      b := u * 2
7      c := v * b
8      v := 3
9      jump L3
10     L2:
11     w := u + v
12     a := w * v
13     v := a - 3
14     if a < 5 goto L3
15     jump L1
16 L3:
17     w := u * v
```

11. Desenați graful fluxului de control și aplicați repetat optimizări locale și propagarea globală a constantelor, până când nu mai este posibil. Doar *w* este *live* la final.

12. Independent de alte optimizări, menționați variabilele *live* în fiecare punct al secvenței. Doar *w* este *live* la final. Propuneți o alocare cu numărul *minim* de registre necesare pentru evitarea *spilling*-ului.