1.Sa se scrie un program pentru implementarea algoritmului de analiza sintactica Earley. Programul primeste la intrare elementele unei gramatici independente de context oarecare, inclusiv cu λ-productii. Programul accepta un numar oarecare de siruri peste alfabetul terminalilor. Pentru fiecare sir se creeaza si se afiseaza tabelele Earley corespondente si daca sirul apartine limbajului generat de gramatica, afiseaza derivarile acelui sir plecand din simbolul de start.

2. Sa se scrie un program care primeste la intrare elementele unei gramatici independente de context oarecare, G, inclusiv cu λ-productii pentru care:
a) calculeaza multimile First(X), Follow(X), pentru fiecare simbol X terminal sau neterminal
b) elimina recursivitatea la stanga (pentru gramatici fără λ-producții)
c) factorizează stânga gramatica G.

3.Sa se scrie un program care implementeaza algoritmul pentru gramatici LL(k) tari. Programul primeste la intrare: k>=1, elementele unei gramatici independente de context, nerecursiva la stanga, oarecare. Programul determina tabela de analiza sintactica asociata si decide daca gramatica data este LL(k). In caz afirmativ, programul permite citirea unui nr oarecare de siruri peste alfabetul terminalilor. Pentru fiecare sir terminal se determina, pe baza tabelei de analiza sintactica obtinuta, daca este in limbajul generat de gramatica respectiva iar in caz afirmativ se afiseaza derivarea sa stanga (o succesiune de numere, fiecare numar reprezintand numarul productiei aplicate) (pot face echipa 2 persoane)

4. Sa se scrie un program care implementeaza algoritmul pentru gramatici SLR(1). Programul primeste la intrare elementele unei gramatici independente de context oarecare. Programul determina tabela de analiza sintactica asociata si decide daca gramatica data este SLR(1). In caz afirmativ, programul permite citirea unui nr oarecare de siruri peste alfabetul terminalilor. Pentru fiecare sir terminal se determina, pe baza tabelei de analiza sintactica obtinuta, daca este in limbajul generat de gramatica respectiva iar in caz afirmativ se afiseaza derivarea sa dreapta (o succesiune de numere, fiecare numar reprezintand numarul productiei aplicate). (pot face echipa 2 persoane)

5. Sa se scrie un program care implementeaza algoritmul pentru gramatici LR(1). Programul primeste la intrare elementele unei gramatici independente de context oarecare. Programul determina tabela de analiza sintactica asociata si decide daca gramatica data este LR(1). In caz afirmativ, programul permite citirea unui nr oarecare de siruri peste alfabetul terminalilor. Pentru fiecare sir terminal se determina, pe baza tabelei de analiza sintactica obtinuta, daca este in limbajul

generat de gramatica respectiva iar in caz afirmativ se afiseaza derivarea sa dreapta (o succesiune de numere, fiecare numar reprezintand numarul productiei aplicate) (pot face echipa 2 persoane).

6. Sa se studieze specificatia pentru generatorul de parser-e Bison. Sa se exemplifice pentru gramatica sintaxei unui limbaj/sublimbaj (se vor considera minim două tipuri de variabile, minim 2 instrucțiuni, dintre care o instrucțiune if și una de ciclare) pentru C++.

7. Sa se studieze specificatia pentru generatorul de parser-e Bison. Sa se exemplifice pentru gramatica sintaxei unui limbaj/sublimbaj (se vor considera minim două tipuri de variabile, minim 2 instrucțiuni, dintre care o instrucțiune if și una de ciclare) pentru Java.

8. Sa se studieze specificatia pentru generatorul de parser-e Bison. Sa se exemplifice pentru gramatica sintaxei unui limbaj/sublimbaj (se vor considera minim două tipuri de variabile, minim 2 instrucțiuni, dintre care o instrucțiune if și una de ciclare) pentru Python.

9. Sa se studieze specificatia pentru generatorul de parser-e Bison. Sa se exemplifice pentru gramatica sintaxei unui limbaj/sublimbaj (se vor considera minim două tipuri de variabile, minim 2 instrucțiuni, dintre care o instrucțiune if și una de ciclare) pentru Prolog sau Haskell.

10. Sa se scrie un program care genereaza un fisier text care va contine functiile (in C sau C++) pentru algoritmul recursiv descendent pentru o gramatica data. La intrare programul primeste gramatica respectiva, care se presupune ca este de tip LL(1). Pentru fiecare productie A->x trebuie sa calculati multimea First(xFolow(A)).