Q1: Answer the following:

A: What are the possibilities in predictive parsing method list them and draw the diagram that illustrates the function of predictive parser (4 marks).

B: Write construction algorithm of predictive parsing table (3 marks).

Q2: Find the First &Follow from the following Grammar: (4 marks)

S→A

A→BC│DBC

B→bB’│^

B’→bB’│^

C→c│^

D→a│d

Q3: Eliminate **Left Recursion** from the following Grammar (4 marks) :

A→A-B│B

B→B÷C│C

C→{A}│id

Q4: Parse the input **bbdccaecb** in the following grammar by using **Shift Reduced** parsing method (12 marks):

**S→AaB**

**A→bAc│d**

**B→eBb│c**

Q5: Consider the following Grammar, build a predictive parsing table: (14 marks)

For string”abbgf

S→A

A→Ad│aB

B→bBC│f

C→g

**Problem-01:**

Consider the following grammar and eliminate left recursion-

A → ABd / Aa / a

B → Be / b

**Solution-**

The grammar after eliminating left recursion is-

A → aA’

A’ → BdA’ / aA’ / ∈

B → bB’

B’ → eB’ / ∈

**Problem-02:**

Consider the following grammar and eliminate left recursion-

E → E + E / E x E / a

**Solution-**

The grammar after eliminating left recursion is-

E → aA

A → +EA / xEA / ∈

**Problem-03:**

Consider the following grammar and eliminate left recursion-

E → E + T / T

T → T x F / F

F → id

**Solution-**

The grammar after eliminating left recursion is-

E → TE’

E’ → +TE’ / ∈

T → FT’

T’ → xFT’ / ∈

F → id

**Problem-04:**

Consider the following grammar and eliminate left recursion-

S → (L) / a

L → L , S / S

**Solution-**

The grammar after eliminating left recursion is-

S → (L) / a

L → SL’

L’ → ,SL’ / ∈

**Problem-05:**

Consider the following grammar and eliminate left recursion-

S → S0S1S / 01

**Solution-**

The grammar after eliminating left recursion is-

S → 01A

A → 0S1SA / ∈

**Problem-06:**

Consider the following grammar and eliminate left recursion-

S → A

A → Ad / Ae / aB / ac

B → bBc / f

**Solution-**

The grammar after eliminating left recursion is-

S → A

A → aBA’ / acA’

A’ → dA’ / eA’ / ∈

B → bBc / f

**Problem-07:**

Consider the following grammar and eliminate left recursion-

A → AAα / β

**Solution-**

The grammar after eliminating left recursion is-

A → βA’

A’ → AαA’ / ∈

**Problem-08:**

Consider the following grammar and eliminate left recursion-

A → Ba / Aa / c

B → Bb / Ab / d

**Solution-**

This is a case of indirect left recursion.

**Step-01:**

First let us eliminate left recursion from A → Ba / Aa / c

Eliminating left recursion from here, we get-

A → BaA’ / cA’

A’ → aA’ / ∈

Now, given grammar becomes-

A → BaA’ / cA’

A’ → aA’ / ∈

B → Bb / Ab / d

**Step-02:**

Substituting the productions of A in B → Ab, we get the following grammar-

A → BaA’ / cA’

A’ → aA’ / ∈

B → Bb / BaA’b / cA’b / d

**Step-03:**

Now, eliminating left recursion from the productions of B, we get the following grammar-

A → BaA’ / cA’

A’ → aA’ / ∈

B → cA’bB’ / dB’

B’ → bB’ / aA’bB’ / ∈

This is the final grammar after eliminating left recursion.

**Problem-09:**

Consider the following grammar and eliminate left recursion-

X → XSb / Sa / b

S → Sb / Xa / a

**Solution-**

This is a case of indirect left recursion.

**Step-01:**

First let us eliminate left recursion from X → XSb / Sa / b

Eliminating left recursion from here, we get-

X → SaX’ / bX’

X’ → SbX’ / ∈

Now, given grammar becomes-

X → SaX’ / bX’

X’ → SbX’ / ∈

S → Sb / Xa / a

**Step-02:**

Substituting the productions of X in S → Xa, we get the following grammar-

X → SaX’ / bX’

X’ → SbX’ / ∈

S → Sb / SaX’a / bX’a / a

**Step-03:**

Now, eliminating left recursion from the productions of S, we get the following grammar-

X → SaX’ / bX’

X’ → SbX’ / ∈

S → bX’aS’ / aS’

S’ → bS’ / aX’aS’ / ∈

This is the final grammar after eliminating left recursion.

**Problem-10:**

Consider the following grammar and eliminate left recursion-

S → Aa / b

A → Ac / Sd / ∈

**Solution-**

This is a case of indirect left recursion.

**Step-01:**

First let us eliminate left recursion from S → Aa / b

This is already free from left recursion.

**Step-02:**

Substituting the productions of S in A → Sd, we get the following grammar-

S → Aa / b

A → Ac / Aad / bd / ∈

**Step-03:**

Now, eliminating left recursion from the productions of A, we get the following grammar-

S → Aa / b

A → bdA’ / A’

A’ → cA’ / adA’ / ∈

**Problem-01:**

Do left factoring in the following grammar-

S → iEtS / iEtSeS / a

E → b

**Solution-**

The left factored grammar is-

S → iEtSS’ / a

S’ → eS / ∈

E → b

**Problem-02:**

Do left factoring in the following grammar-

A → aAB / aBc / aAc

**Solution-**

**Step-01:**

A → aA’

A’ → AB / Bc / Ac

Again, this is a grammar with common prefixes.

**Step-02:**

A → aA’

A’ → AD / Bc

D → B / c

This is a left factored grammar.

**Problem-03:**

Do left factoring in the following grammar-

S → bSSaaS / bSSaSb / bSb / a

**Solution-**

**Step-01:**

S → bSS’ / a

S’ → SaaS / SaSb / b

Again, this is a grammar with common prefixes.

**Step-02:**

S → bSS’ / a

S’ → SaA / b

A → aS / Sb

This is a left factored grammar.

**Problem-04:**

Do left factoring in the following grammar-

S → aSSbS / aSaSb / abb / b

**Solution-**

**Step-01:**

S → aS’ / b

S’ → SSbS / SaSb / bb

Again, this is a grammar with common prefixes.

**Step-02:**

S → aS’ / b

S’ → SA / bb

A → SbS / aSb

This is a left factored grammar.

**Problem-05:**

Do left factoring in the following grammar-

S → a / ab / abc / abcd

**Solution-**

**Step-01:**

S → aS’

S’ → b / bc / bcd / ∈

Again, this is a grammar with common prefixes.

**Step-02:**

S → aS’

S’ → bA / ∈

A → c / cd / ∈

Again, this is a grammar with common prefixes.

**Step-03:**

S → aS’

S’ → bA / ∈

A → cB / ∈

B → d / ∈

This is a left factored grammar.

**Problem-06:**

Do left factoring in the following grammar-

S → aAd / aB

A → a / ab

B → ccd / ddc

**Solution-**

The left factored grammar is-

S → aS’

S’ → Ad / B

A → aA’

A’ → b / ∈

B → ccd / ddc