1)

a) FIRST (S) = { b }
FIRST (A) = { c , d }

b) FIRST (bab) = { b }
FIRST (bA) = { b }

FIRST (d) = { d }
FIRST (cA} = { c }

c) Yes there is an overlap with FIRST (bab) and FIRST (bA)

d) No the grammar is not left-recursive.

e) Grammar is not LL(1) since both the productions of S has b as the first terminal.

2)

a) FIRST (S) = { b }
FIRST (C) = { a , d , c }
FIRST (A) = { d , c }

b) FIRST (bC) = { b }

FIRST (ab) = { a }
FIRST (A) = { d , c }

FIRST (d) = { d }
FIRST (cA) = { c }

c) No there is no overlap

d) No the grammar is not left-recursive.

e) Grammar is LL(1) since it is not left-recursive and there is no overlap in the productions FIRST SETS

3)

a) FIRST (S) = { b , . }
FIRST (A) = { b , . }
FIRST (B) = { c }
FIRST (C) = { b , . }

b) FIRST (AB) = { b , . }

FIRST (Ca) = { b }
FIRST (ε) = { . }

FIRST (BaAC) = { c }
FIRST (c) = { c }

FIRST (b) = { b }
FIRST (ε) = { . }

c) Yes there is an overlap between the FIRST sets of two right-hand sides of the same non-terminal(for nonterminal B).

d) Yes the grammar is left-recursive.

e) The grammar is not LL(1) since it is left-recursive and has an overlap.

4)

a) FIRST (S) = { b , . }
FIRST (A) = { b , . }
FIRST (B) = { c }
FIRST (D) = { a , . }
FIRST (C) = { b , . }

b) FIRST (AB) = { b , . }

FIRST (Ca) = { b , . }
FIRST (ε) = { . }

FIRST (cD) = { c }

FIRST (aAcD) = { a }
FIRST (ε) = { . }

FIRST (b) = { b }
FIRST (ε) = { . }

c) Yes there is an overlap between the FIRST sets of two right-hand sides of the same non-terminal (for nonterminal A).

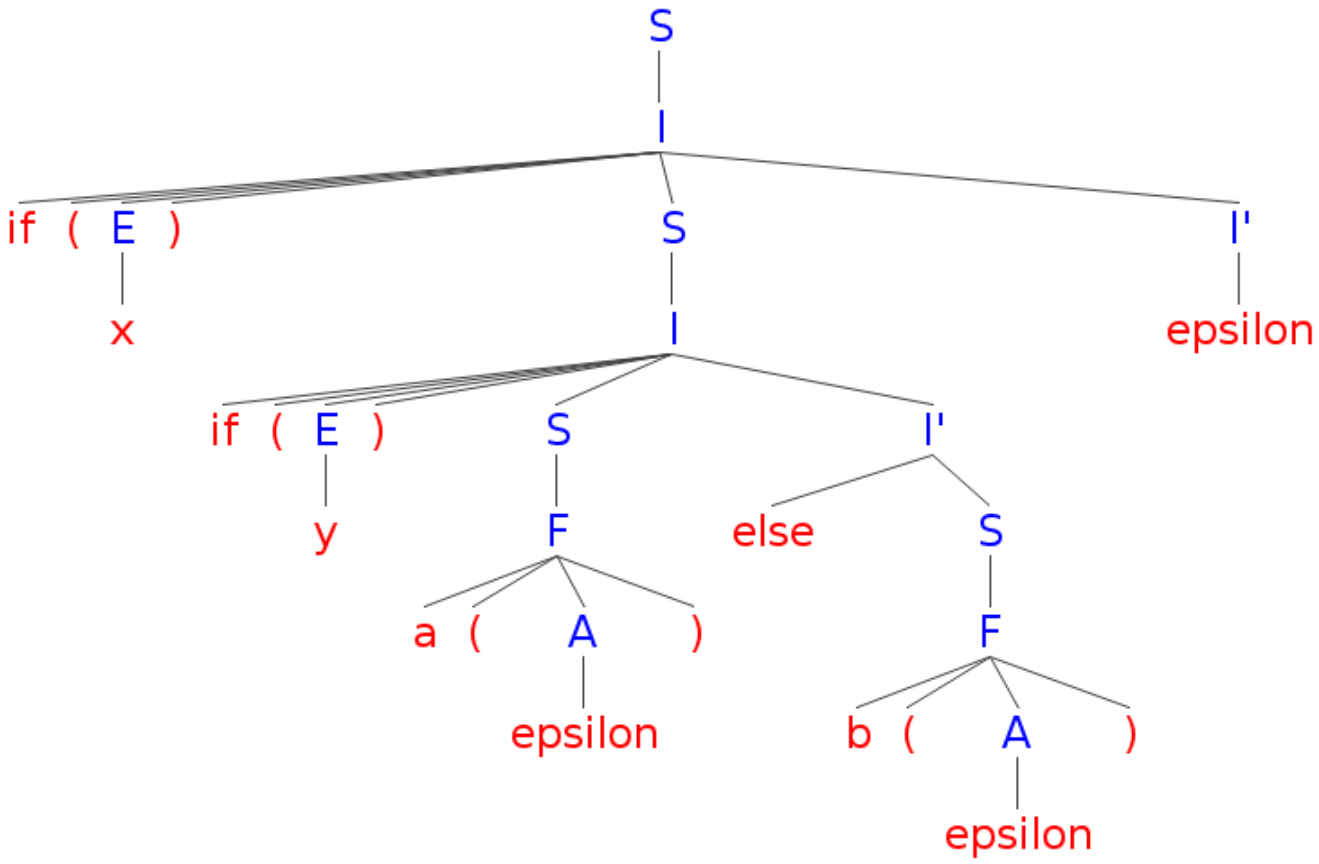
d) No the grammar is not left-recursive.

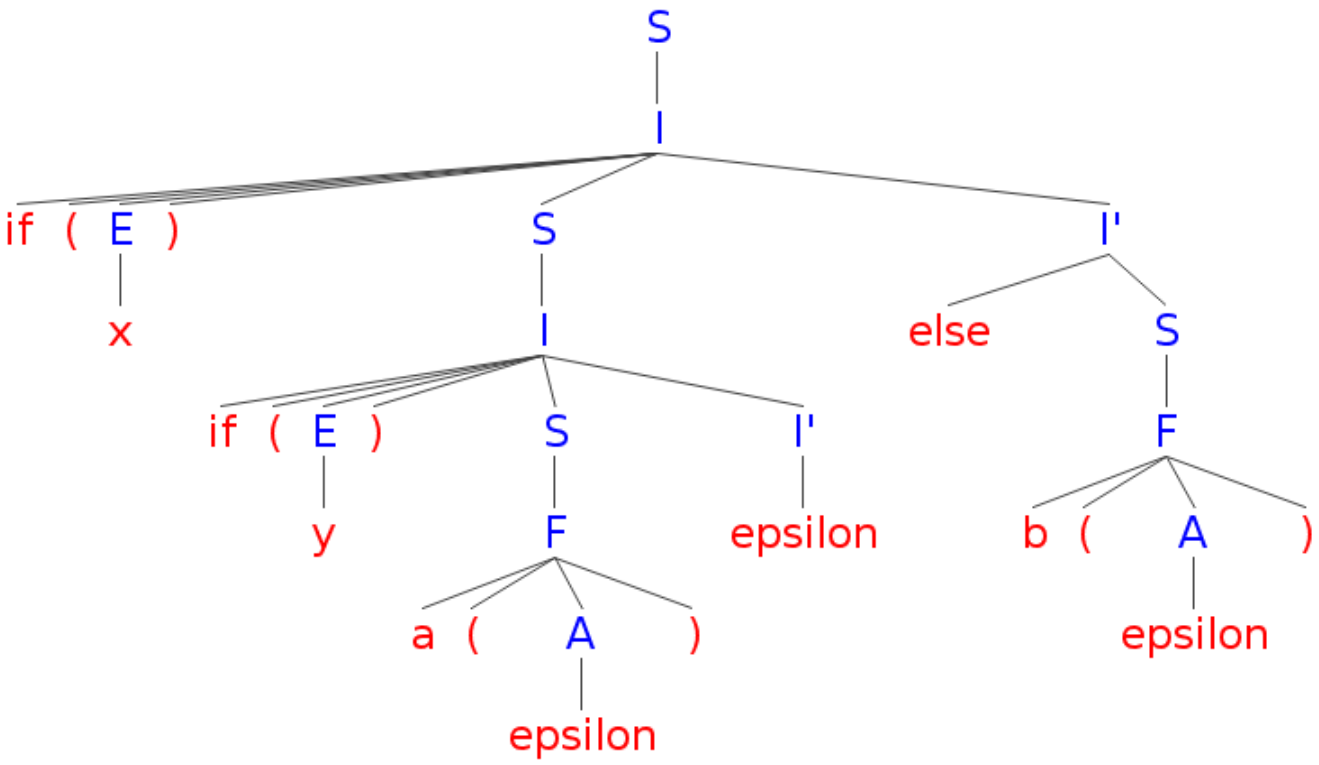
e) Yes the language is LL(1) even though there is an overflow because they are epsilon overlaps and the follow sets don't have intersections

5)

S → F | I
F → IDENT (A)
I → if (E) S I’
I’ → else S | ε
E → IDENT
A →IDENT A’ | ε
A’ → , A | ε

6)





7)

```
let rec parseL toklis = parseLprime (parseE toklis)

and parseLprime toklis = match (hd toklis) with
                         | SEMIC → parseL ( tl toklis )
                         | _     → toklis
and parseE toklis = match (hd toklis) with
                    | ID(s) | INT(i) → tl toklis
                    | _              → raise SyntaxError
```