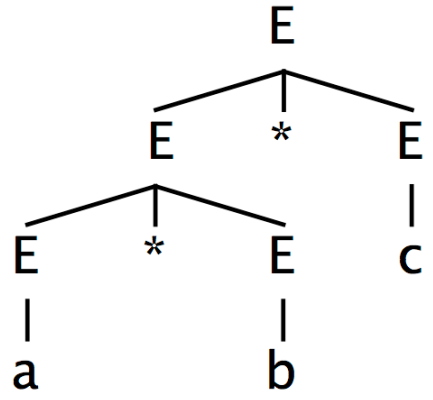
Problem 1)

```
let rec cst_to_ast t = match t with
| E1(t1, token, t2) -> (match token with
                        | Minus -> Sub(cst_to_ast t1, cst_to_ast t2))

| E2(t) -> cst_to_ast(t)

| T1(tok) -> (match tok with
              | Ident(str) -> Id(str))

| T2(tree, token1, token2) -> (match (token1, token2) with
                               | (Star, Ident(str2)) -> Times(cst_to_ast tree, Id(str2)))
```

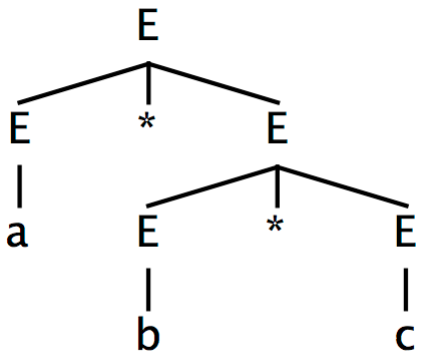
Problem 2)

A)

1)



2)



B)

Shift-reduce Parse for Tree 1

Action	Stack	Input
Shift		a*b*c
Reduce	a	*b*c
Reduce	E	*b*c
Shift	E*	b*c
Reduce	E*b	*c
Reduce	E*E	*c
Shift	E	*c
Shift	E*	c
Reduce	E*c	eof
Reduce	E*E	eof
Accept	E	eof

Shift-reduce Parse for Tree 2

Action	Stack	Input
Shift		a*b*c
Reduce	a	*b*c
Reduce	E	*b*c
Shift	E*	b*c
Reduce	E*b	*c
Shift	E*E	*c
Shift	E*E*	c
Reduce	E*E*c	eof
Reduce	E*E*E	eof
Reduce	E*E	eof
Accept	E	eof

C)

I) The first parse tree is correct

II) The first shift reduce parse is correct since it takes the right step at Step 6 of reducing

D)

%left Minus
%left Star

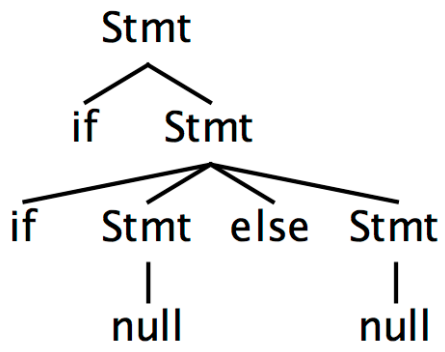
Problem 3)

Extended grammar will be:

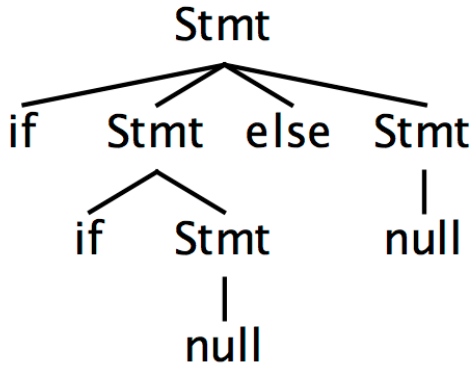
E -> T=E | T
T -> T==P | P
P -> P-O | O
O -> O*id | id

Problem 4)

A) The Correct Parse Tree



Incorrect Parse Tree



B)

Correct Shift Reduce Parse

Action	Stack	Input
Shift		if if null else null
Shift	if	if null else null
Shift	if if	null else null
Reduce	if if null	else null
Shift	if if Stmt	else null
Shift	if if Stmt else	null
Reduce	if if Stmt else null	eof
Reduce	if if Stmt else Stmt	eof
Reduce	if Stmt	eof
Accept	Stmt	eof

Incorrect Shift Reduce Parse

Action	Stack	Input
Shift		if if null else null
Shift	if	if null else null
Shift	if if	null else null
Reduce	if if null	else null
Reduce	if if Stmt	else null
Shift	if Stmt	else null
Shift	if Stmt else	null
Reduce	if Stmt else null	eof
Reduce	if Stmt else Stmt	eof
Accept	Stmt	eof

C)
%right if
%nonassoc else

since else has a greater precedence than if.