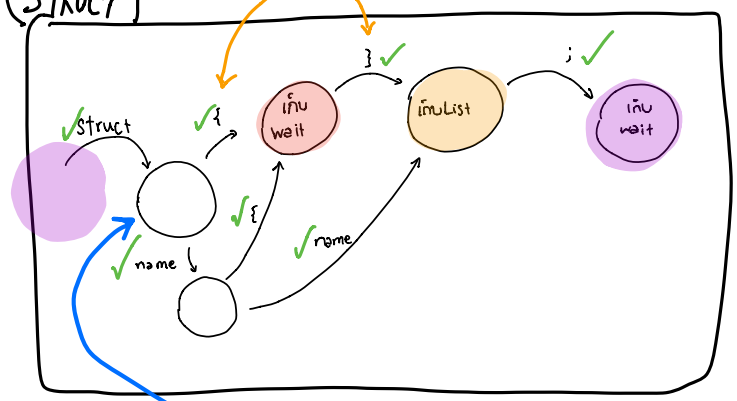
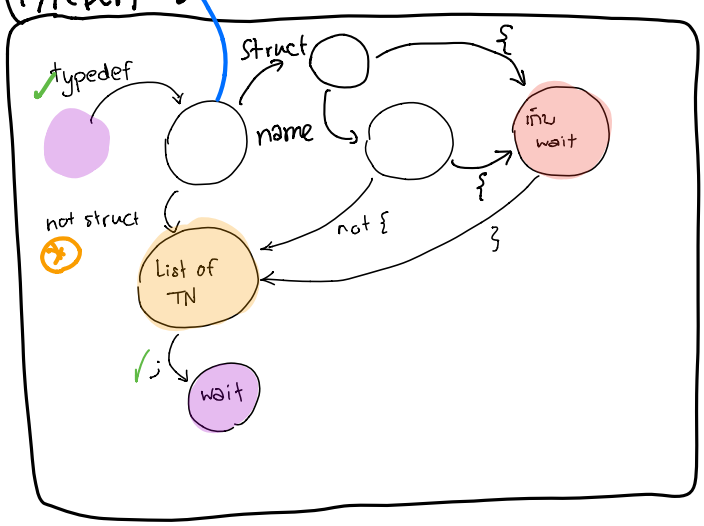
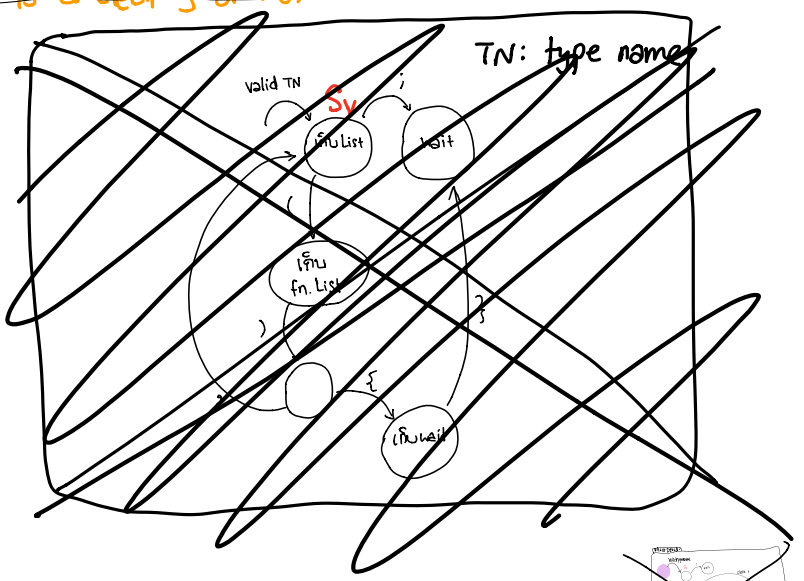
No ~~we need {} handler → Return if } is a clear } or not~~

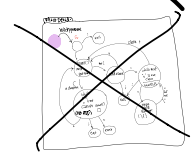
STRUCT



TYPEDEF







GET TOKEN
→ TOKEN

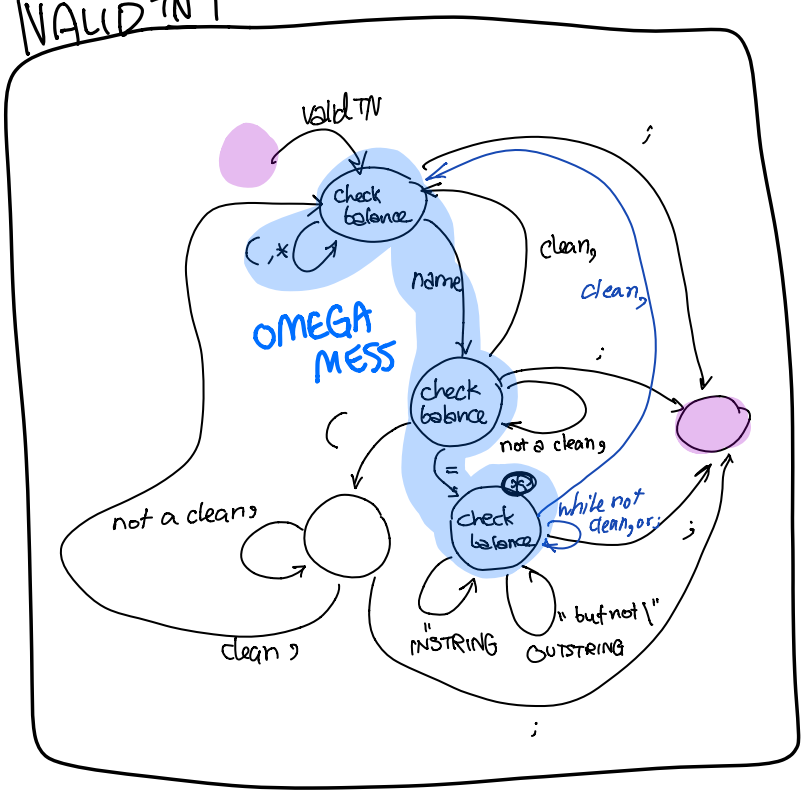
1 UNIVERSAL FUNCTION

gettoken
- → typedef ✓
- → struct ✓
- → valid TN
- → NAME ✓
- → { ✓
- → } ✓
- → ; ✓
- → ,
- → (✓
- →) ✓
- → [✓
- →] ✓
- → "
- → \"
- → STRING CONST.
- → [NUMBER(floats)] or . , - , + , *

FUNCTIONALITY
→ SKIP THROUGH COMMENTS
→ SKIP WHITE SPACES

COMMENT SKIPPER

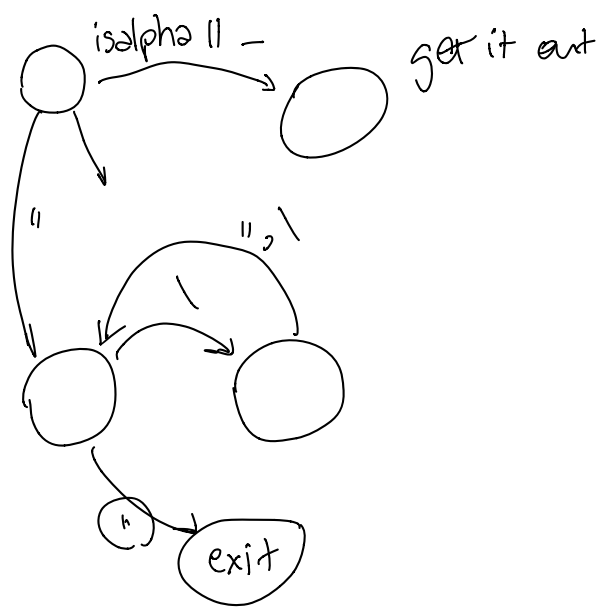
VALID TN

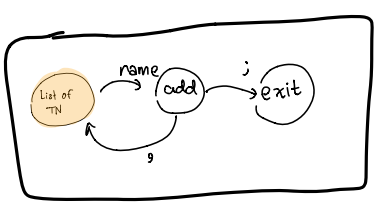


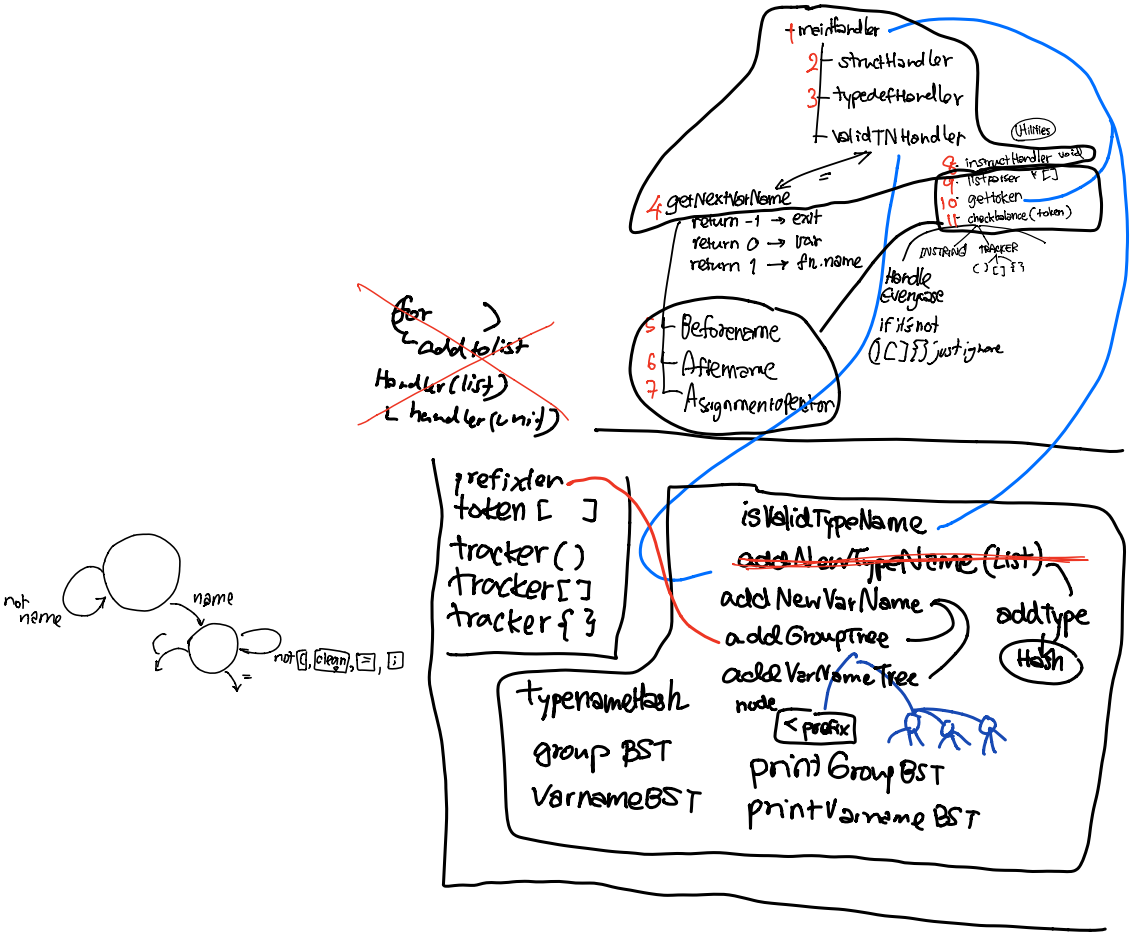
BraceChecker

check balance of (), [], {}

return 0 if balanced





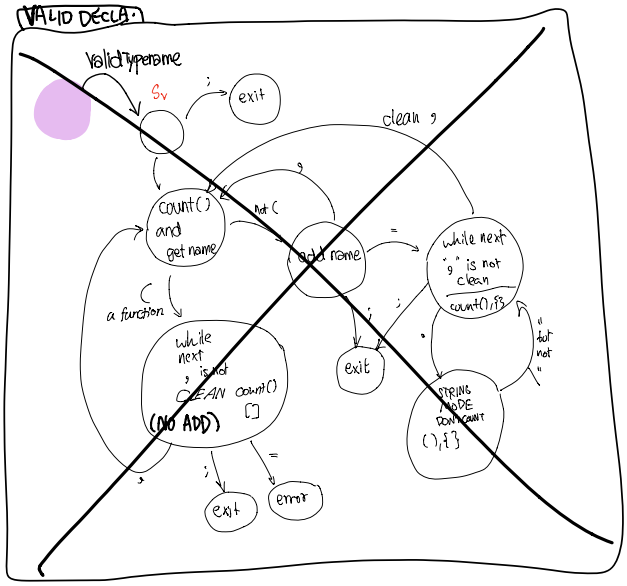


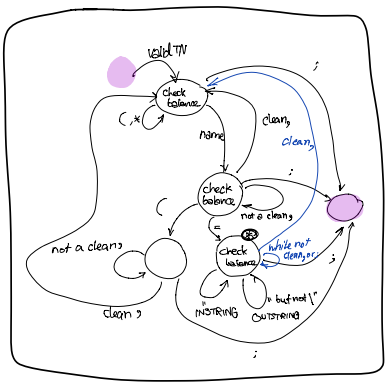
Structs

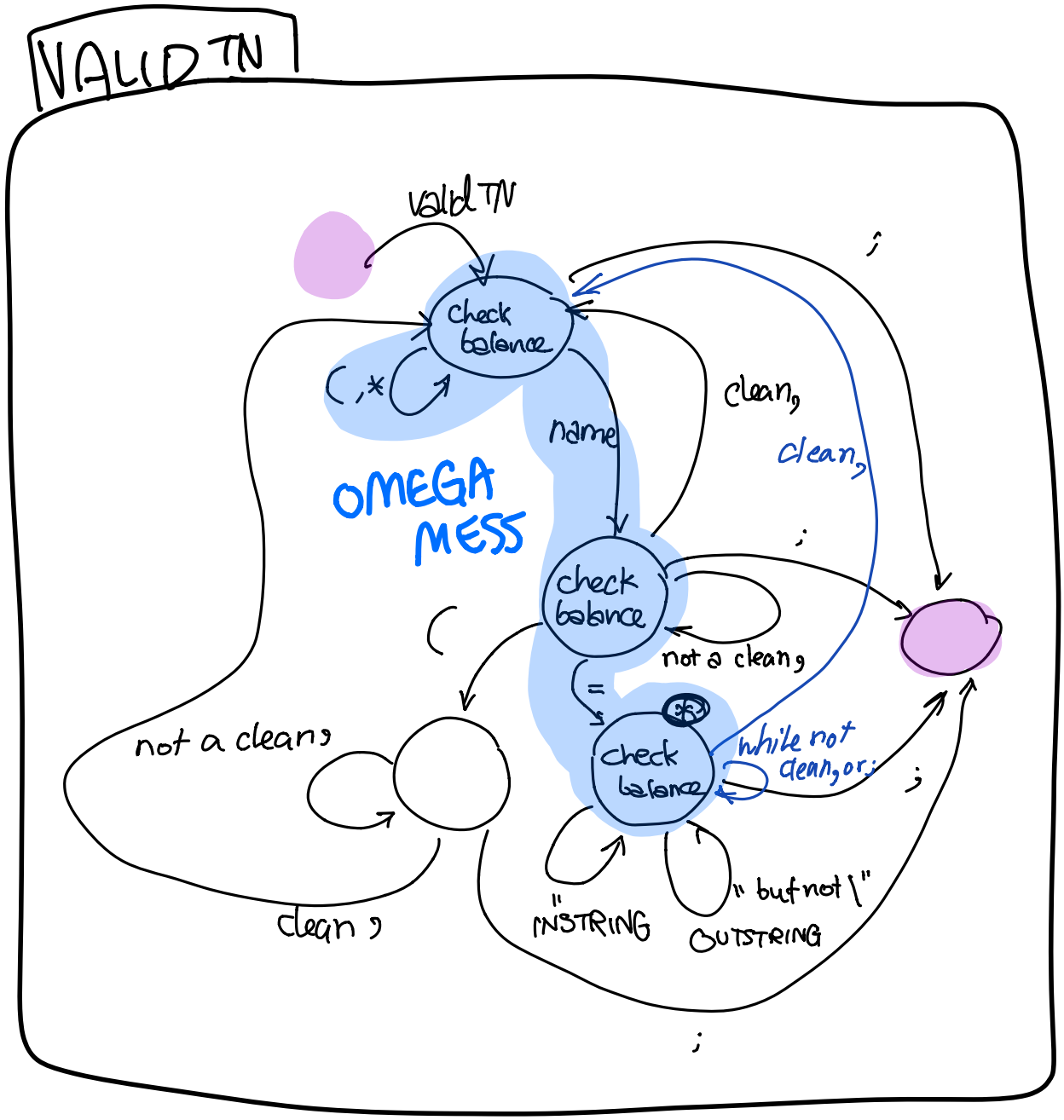
```
int a, *b, c[2] = {0,3}, (*(*d(char *p, int k))[])(), e ;

typedef to t1,t2,t3;

typedef struct t1 t2,t3,t4,t5;

typedef struct t1 { } t2,t3,t4;
```

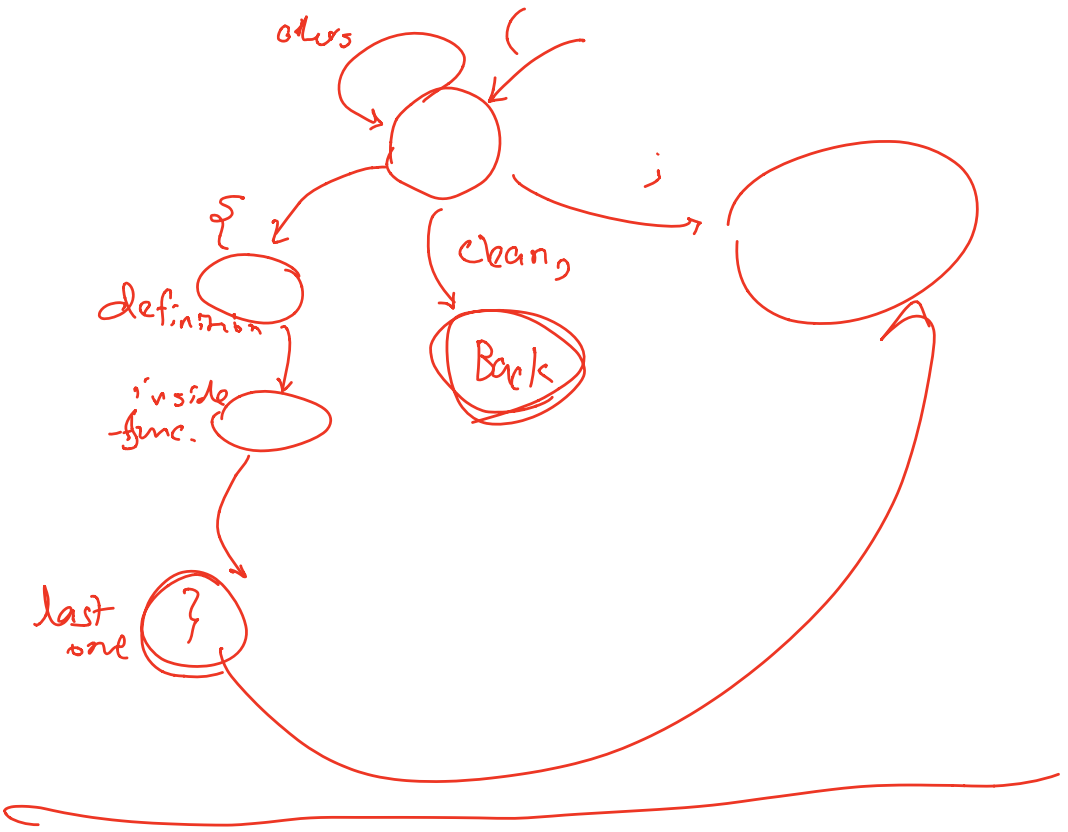
```
/* Not doing */

printf("int a");

char a[ ] = " char x;" ;
```

```
char *p[ ] = { "char x",

             "char y",

             "char z" } ;
```


VALID DECLA.
ValidTypename
S_v
;
exit
clean ,
count() and get name
not (
add name
=
while next "," is not clean
count(),{}
a function
while next , is not CLEAN count() []
(NO ADD)
exit
STRING MODE DONT COUNT (),{}
" but not "
exit
error


valid TN
check balance
(,*
name
clean,
clean,
check balance
not a clean,
not a clean,
check balance
while not clean, or ;
clean ,
"INSTRING
OUTSTRING
" but not \"


VALID TN
valid TN
check balance
(,*
name
clean,
clean,
;
OMEGA MESS
check balance
not a clean,
=
check balance
while not clean, or ;
;
not a clean,
clean,
"INSTRING
" but not \"
OUTSTRING
;



Block Handler

{

}

function definition

int main( ){

return

}

struct{

}

{

}