SPARSH AGARWAL
CS 536
HW 7
9075905142

A1) The extra productions needed are —
(assuming that id is a non terminal used for
production to ID)

1) type → id
2) typedef → TYPEDEF STRUCT type id id SEMICOLON
3) structDecl → id LCURLY structBody RCURLY SEMICOLON

A2) @ (i) a flag to check if it is a type or a variable

A3) The symbol table will be.

	name	type	flag	link
①	MonthDayYear	struct	False	
②	date	MonthDayYear	True	MonthDayYear
③	today	MonthDayYear	False	date
④	dollars	int	True	
⑤	salary	int	False	dollars
⑥	moreDollars	int	True	dollars
⑦	nd	int	False	moreDollars
⑧	d	int	False	

A2) (a)(i) a flag to check if it is a type or a variable

(ii) a link to connect it to its type or id from which it is created

(iii) also maintain a type variable of in all alias to specify & pass on their type to children.

(b) → first xxx should be check for duplicate declarations,

→ then if xxx has not been declared before, T should be checked by lookupglobal for latest declaration & its type should be determined, also whether it is itself a type or alias or variable should be determined.

→ then if T is a type alias & 'xxx' is declared for first time, create a new symbol with type similar to T & create a link of xxx to T; flag of type is true since this is a type alias.

ⓒ To process a declaration of variable, function or parameter named 'xxx' with type T.

→ first check for duplicate declarations of xxx

→ then check if T is a valid type (built-in type, struct type or check by lookupglobal if its latest declaration has flag of typealias true)

→ if T is valid type & 'xxx' is unique, create a new symbol in symbol table with ~~variable~~ name 'xxx' & type T & put type of T in type of 'xxx' also the flag of typealias will be false since this is not a typedef, also link it with its parent node.

ⓓ To process xxx in a statement, check its flag whether is a type alias or variable & then handle it accordingly using its type field & link to parent symbol.