LL(1) parsing table

Non tirmanal	First	Follow
Program	#oil, key, engine, ignite, ε	$
ImportSection	#oil, ε	key, engine, ignite, $
Import	#oil	
NamespaceSection	key	
FunctionList	engine, ignite, ε	$
Function	engine, ignite	
ParamList	gear, track, exhaust, turbo, flag, ε	)
Param	gear, track, exhaust, turbo, flag	
Block	{	engine, ignite, $, }, pitstop
StatementList	gear, track, exhaust, turbo, flag, Identifier, announce, finishline, ε	}
Statement	gear, track, exhaust, turbo, flag, Identifier, announce, finishline	
Expression	Identifier, Number, String, (	;,), pitstop
Term	Identifier, Number, String, (	
Factor	Identifier, Number, String, (	
Type	gear, track, exhaust, turbo, flag	Identifier

Non-Terminal	Input Token	Production Used
Program	#oil	Program → ImportSection NamespaceSection FunctionList
Program	key	Program → ImportSection NamespaceSection FunctionList
Program	engine	Program → ImportSection NamespaceSection FunctionList
ImportSection	#oil	ImportSection → Import ImportSection
ImportSection	key	ImportSection → ε
NamespaceSection	key	NamespaceSection → key Identifier
FunctionList	engine	FunctionList → Function FunctionList
FunctionList	ignite	FunctionList → Function FunctionList
FunctionList	$	FunctionList → ε
Function	engine	Function → engine Identifier (ParamList) Block
Function	ignite	Function → ignite (ParamList) Block
Type	gear	Type → gear
Type	track	Type → track
Type	exhaust	Type → exhaust
Type	turbo	Type → turbo
Type	flag	Type → flag
Statement	announce	Statement → announce Expression ;
Statement	finishline	Statement → finishline Expression ;
Statement	Identifier	Statement → Identifier = Expression ;
Statement	gear	Statement → Type Identifier = Expression ;
Statement	track	Statement → track (Expression) Block pitstop Block
Statement	looplap	Statement → looplap (Expression) Block
Statement	listen	Statement → listen Identifier ;