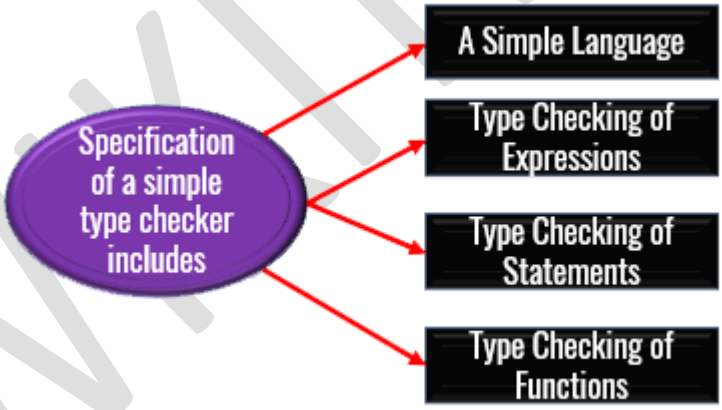
Specification of a Simple Type Checker

- Specification of a simple type checker for a simple language in which the type of each identifier must be declared before the identifier is used.
- The type checker is a translation scheme that synthesizes the type of each expression from the types of its subexpressions.
- The type checker can handle arrays, pointers, statements, and functions.
- Specification of a simple type checker includes the following:
    - A Simple Language
    - Type Checking of Expressions
    - Type Checking of Statements
    - Type Checking of Functions



Simple Language

- The following grammar generates programs, represented by the nonterminal P, consisting of a sequence of declarations D followed by a single expression E.

 © Copyright 2016. All Rights Reserved.

P -> D ; E
D -> D ; D | id : T
T -> char | integer | array[num] of T | ‡ T

A translation scheme for above rules:

P -> D ; E
D -> D ; D
D -> id : T { addtype(id.entry, T.type }
T -> char { T.type := char }
T -> integer { T.type := integer }
T -> ‡ T1 { T.type := pointer(T1.type) }
T -> array[num] of T1 { T.type := array(1..num.val, T1.type) }

For More Details Click Here:

https://www.wikitechy.com/tutorials/compiler-design/specification-of-a-simple-type-checker

WIKITECHY

 © Copyright 2016. All Rights Reserved.