CSE 420

13 February 2024

Java, C, Python hybrid compiler use করে,

Lexical analysis phase এ lexical error হয়, যদি token generate না হয়,

↳ Panic mode recovery (lexical error recover solve করার এক উপায়)

> # token অনুযায়ী gen করে grammar আগে base এ define করতে হবে.

if (a = = b)
① ② ③ ④ ⑤ ⑥

no of lexemes: 6

no " tokens: 6 → যে কয়টা ~~token~~ lexeme ঠিক ততটা token generate হবে,

a = b + e

if ( ) ↳ ২টা আলাদা lexeme.

int sum
ID

$(a-b)^+$
↳ a
↳ b
↳ aa
↳ bb
↳ aab
↳ bab ...

regex — { }
if { token = IF }.
({ token = (}
== { token = ADDOP }
) { token =) }
+ { token = + }
$\{(a-b)^+\}$ { token = ID }

$\{(a-z)^+\}$ = ~~fi~~

lexical error.

fi + (a = = b)

lexical analyzer token generate korar somoy.

SDT use হয়

syntax directed translation

> # Semantic analysis এ logical error checking হয়, variables are declared or not, scoping ... check in the semantic analysis

if (a = = b)

IF (ID ADDOP ID)

value গুলো syntax analyzer stack এ push করতে থাকে,

fi + (a = = b)

fi + lexeme টা কারো সাথে match করছে না.

fi (a==b).

ID (ID ADDOP ID) → ভুল code, এখানে ID থাকার কথা না।

fi এর জন্য Identifier token generate হয়েছে as তার last pattern {(a-z)+} এর সাথে match করেছে।

→ Syntax analysis phase এ গিয়ে এই ভুল ধরা পড়বে। lexical analysis এ dhora poreni as se successfully token gen করেছে, syntax analysis এ ধরা পড়বে কারণ সে grammar অনুযায়ী check করে যে সেখানে সে IF (ID ADDOP ID) হওয়ার কথা ছিল তা পায়নি।

fi + (a==b)

fi+ lexeme টা কারো সাথে match করেনি তাই token generate করতে পারেনি so lexical error.

ekhon lexical error এর জন্য যে recovery method design kore dekhi seta use করব।

as lexical error এখন যে panic mode recovery active করব।

panic mode.

pattern match করা।

ektu ektu kore character বাদ দেয়া, start করবে।
ekhon or concern sesh হয় token generate করে দিচ্ছে।

Parsing, scanning সব usually left to right হয়।

error কোনো stage এ ধরা না পড়লে grammar (CFG). ঠিকমতো define করা হয়নি।

{ ~~eif~~ {token = ID}.

$\{(a-z)^+\}$ ↗

~~e~~if (a == b)

IF

successfully panic mode recover kaj করা as eif আগে থেকে match koreche না, so ekhn se e বাদ দিয়ে এখন, if pattern এর সাথে match করছে, so successfully panic mode recovery.

token generate না হলে, panic mode এ token generate না হলেও পরবর্তী আর কোনো কাজ হবে না।

panic mode use করে token generate হয়, তবে tracking করে...

সব corner case চিন্তা করে pattern design করবে।

Syntax error:

- ⇒ missing or misplaced punctuations
- ⇒ incorrect use of keywords. (elf : if)
- ⇒ mismatched brackets.
- ⇒ incorrectly structured statements.

} কি কি কারণে syntax error arise হয়।

a = = b
a = b

↳ `if (a=b);` → incorrectly structured statements

grammar কিভাবে define করা এইসব check করছে তার ভিত্তিতে।

প্রত্যেক language difference because তাদের grammar ভিন্নভাবে define করা হয়।

Python ;
C ;

recovery method থেকে detailed question আসে।

Semantic error:

- → Using variables that have not been declared.
- → type mismatches.
- → misuse of operators or functions.
- → logical errors.

- a variable use করছি ; কিন্তু আগে কোথাও define করিনি,

a=4 ; so semantic error ①

- a datatype int ; এখন a তে যদি store করলাম "abc" - string. ∴ type mismatch ②

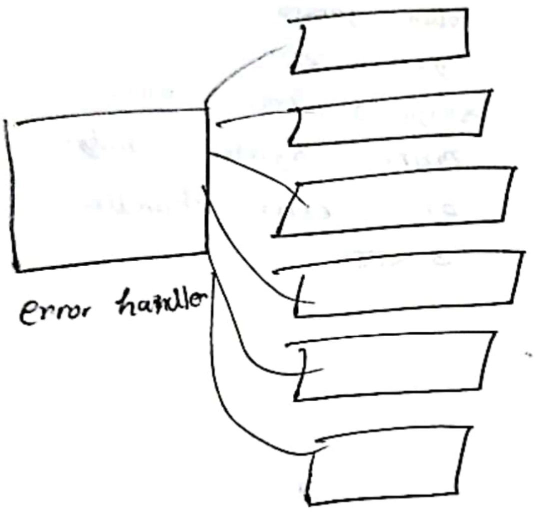
③ z = a++b ; operator বা function হিসাবেও use না করলে।

④ z = a+b হওয়ার কথা ছিল, = a+b

[a+b = z] → logical error.

Error Handler.

6 টা phase ছিল compile.



এরা error handle করে, কত number line এ error আছে; তা track করতে পারে এবং user কে error show করে।

Semantic analysis phase এ error. আবার, error handle করবে. var tomar ei line aer eitor jonno error asche.

traceback করে, lexical analysis stage এ গিয়ে তার input এর সাথে check করে line find করে কোন line এ error show করে।

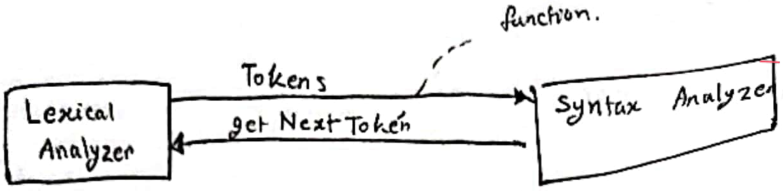
↳ traceback করতে না পারলে error handle করতে তা আলাদাভাবে define করে দিতে হবে।

INT ID

int { }

The role of lexical Analyzer

Token generate করা, পরে ও parser কে pass করে দেয়া.



parallely কাজ করছে। lexer loop এর ভিতর lexeme এর পর আবার token generate করছে। token gen করে। এরকম stage এ কোনো error পেলে syntax analyzer এ error handle এ যাবে

<u>int</u> <u>a;</u>
INT ID SEMI COLON