Errors and its classification

Akshay Singhal Compiler Design

What are errors?

In the programming world, errors are the bugs which may prevent the program to compile and run correctly as per the expectations of the programmer.

Classification of errors:

1. Compile Time Errors
    1. Lexical Phase Errors
    2. Syntactic Phase Errors
    3. Semantic Errors
2. Run Time Errors

1. Compile Time Errors

A) Lexical Phase Errors-

These are those errors which are detected during lexical analysis phase.

Typical lexical phase errors are-

a) Exceeding length of identifier or numeric constants-

Example- If in FORTRAN, there is an identifier whose length is of 10 characters, then lexical error occurs.

b) Appearance of illegal characters-

Example- In the statement printf (“Hello”);$ the appearance of $ at the end causes the lexical error.

c) Unmatched string-

Example- */ gives rise to lexical error because only the end of comment is present but the beginning of the comment is not present.

d) Spelling errors which causes errors in token formation-

Example-

```
swtich (choice)
{
}
```

Here, ‘swtich’ can not be recognized as a misspelling of keyword ‘switch’. Rather, lexical analyzer will understand it as a valid identifier.

Error Recovery-

The most commonly used strategy is panic mode error recovery. In this recovery mechanism, successive characters from the remaining input are deleted until the well formed token is found i.e. if any unwanted character occurs, then it is deleted to recover from error and if any unmatched string occurs then appropriate string / character is inserted in order to match the string.

B) Syntactic Phase Errors-

These are those errors which are popularly known as syntax errors. They are detected during syntax analysis phase of compiler.

Typical lexical phase errors are-

1. Errors in structure
2. Missing operators
3. Misspelled keywords
4. Unbalanced parenthesis

Example-

In the following code, the keyword ‘switch’ is incorrectly written as swtich because of which lexical analyzer considers it as a valid identifier and generates the tokens.

```
swtich (choice)
{
}
```

Now, when the parser during syntactic phase demands for tokens from lexical analyzer and if the tokens do not satisfy the grammatical rules of programming language, then syntactic error gets raised.

Error Recovery-

Parser employs various strategies to recover from syntactic errors. These strategies are-

1. Panic mode
2. Phrase level
3. Error productions
4. Global production

1. Panic mode-

- This is the most commonly used strategy by most parsing methods.
- It is simple to implement and guarantees not to go in infinite loop.
- In this strategy on discovering error, the parser discards input symbol one at a time. This process is continued until one of a designated set of synchronizing tokens is found.
- Thus, in panic mode recovery, a considerable amount of input is skipped without checking it for additional errors.

2. Phrase level recovery-

- In this method on discovering error, parser performs local correction on remaining input.
- It can replace a prefix of remaining input by some string. This actually helps the parser to continue its job.
- The local correction must be-

1. replacing comma by semicolon
2. deletion of extra semicolons
3. inserting missing semicolon

3. Error Production-

- If we have knowledge of common errors that can be encountered, then we can incorporate these errors by augmenting the grammar of the corresponding language with error productions that generate the invalid constructs.
- This method is extremely difficult to maintain because if we change the grammar then it becomes necessary to change the

corresponding error productions.

4. Global Production-

- Global production is simply a theoretical concept. Such methods increase time and space requirements at parsing time.

C) Semantic Errors-

Semantic errors are those errors which get detected during semantic analysis phase.

Typical errors in this phase are-

1. Incompatible types of operands
2. Undeclared variables
3. Not matching of actual arguments with formal arguments

Example-

```
int a[10], b;
a = b ;       // This code generates error
```

Error Recovery-

1. If the error “Undeclared identifier” is caught then to recover from this error, the symbol table entry for corresponding identifier is made.
2. If the data types of two operands in an expression are not compatible with each other then automatic type conversion is done by compiler. This type of conversion is called implicit conversion.

To get more videos, notes and other study material of Compiler Design, click here.

To practice more Previous Years GATE Problems from Compiler Design, click here.

To watch more videos, visit our YouTube channel LearnVidFun.