Error Recovery in Parsing

- A parser should be able to detect and report any error in the program.
- The process of locating errors and reporting it to user is called *Error Handling Process*.
- It is expected that when an error is encountered, the parser should be able to handle it and carry on parsing the rest of the input.
- A parser is mainly responsible for handling of the Compile time errors which may be encountered at various stages of the compilation process.
- A program may have the following kinds of errors at various stages of compilation:
    - Lexical Phase Errors (name of some identifier typed incorrectly)
    - Syntactic Phase Errors (missing semicolon or unbalanced parenthesis)
    - Semantic Errors (incompatible value assignment)

There are four common error recovery strategies that can be implemented in the parser to deal with errors in the code.

1. **Panic Mode:**
    - In this method, when an error is encountered anywhere in the statement, successive characters from the input are ignored one at a time until a designated set of synchronizing tokens is found. Synchronizing tokens are delimiters such as ; or } .
    - The advantage of this method is that it is the easiest to implement and guarantees not to go to infinite loop.
    - The disadvantage is that a considerable amount of input is skipped without checking it for additional errors.

2. **Statement Mode:**
    - In this method, when a parser encounters an error, it performs necessary correction on remaining input so that the rest of input statement allow the parser to parse ahead.
    - The correction can be deletion of extra semicolons, replacing comma by semicolon or inserting missing semicolon.
    - While performing correction, utmost care should be taken for not going in infinite loop.
    - Disadvantage is that it finds difficult to handle situations where actual error occurred before point of detection.

3. **Error Production:**
   - If user has knowledge of common errors that can be encountered then, these errors can be incorporated by augmenting the grammar with error productions that generate erroneous constructs.
   - If this is used then, during parsing appropriate error messages can be generated and parsing can be continued.
   - Disadvantage is that it is difficult to maintain.

4. **Global Correction:**
   - The parser examines the whole program and tries to find out the closest match for it which is error free.
   - The closest match program has a smaller number of insertions, deletions and changes of tokens to recover from erroneous input.
   - Due to high time and space complexity, this method is not implemented practically.

References:

1. https://www.geeksforgeeks.org/error-detection-recovery-compiler/
2. https://www.tutorialspoint.com/compiler_design/compiler_design_error_recovery.htm