Haseeb Ahmed (FA21-BCS-042)

Compiler Construction Lab Terminal Q2

Here’s an explanation of the three core functions from the mini compiler:

**1. checkOperator(caractere)**

* **Purpose:**  
  Identifies and classifies operators in the input source code.
* **Functionality:**  
  This function likely examines the character (caractere) to determine if it matches a known operator, such as +, -, \*, /, =. If it finds a match:
  + It classifies the operator into categories like arithmetic, logical, relational, or assignment operators.
  + The identified operator is then stored as a token in the token stream, along with its type and position in the source code.
* **Example:**  
  Input:
* a + b \* c;

Processing:

* + Encounters + → classifies as an arithmetic operator.
  + Encounters \* → classifies as another arithmetic operator.

**2. checkCommenaireEtEnd(caractere, SnailFile)**

* **Purpose:**  
  Handles comments and determines the end of the file.
* **Functionality:**
  + If the character indicates the start of a comment (e.g., // for single-line comments or /\* for multi-line comments), it skips over the comment until the end (\n or \*/).
  + If the character signals the end of the file (EOF), it stops further processing.
* **Example:**  
  Input:
* int x = 10; // This is a comment

Processing:

* + Encounters // → ignores everything after it on the line.
  + Reaches EOF → signals the lexical analysis to terminate.

**3. checkprintf(caractere, SnailFile)**

* **Purpose:**  
  Detects and processes specific syntax constructs, like the printf statement.
* **Functionality:**
  + Checks if the character sequence matches the printf keyword.
  + If a match is found, it classifies it as a function or keyword token.
  + It may also extract and tokenize the string argument passed to printf, ensuring proper syntax (e.g., balanced quotes).
* **Example:**  
  Input:
* printf("Hello, World!");

Processing:

* + Detects printf → classifies as a keyword token.
  + Tokenizes the string "Hello, World!" and ensures it adheres to the language's syntax rules.

**Integration**

These functions work together to scan the source code, identify meaningful components (tokens), and handle irrelevant parts like comments. This prepares the code for syntactic and semantic analysis in subsequent stages of the compiler.