

Programming Paradigms – 2052

Lab – 03

Section 2- Thinking questions

1. What is the primary purpose of BNF and EBNF in programming language design?

Such outlines help capture the foundational principles (grammar) of a programming language for human users and compilers to give structure to the relevant prog

2. EBNF is shorter and easier to read. It uses symbols like {} for repetition and [] for optional parts.

BNF: `<digits> ::= <digit> | <digit> <digits>`

EBNF: `<digits> ::= <digit> {<digit>}`

3. Why is defining a formal grammar important for compilers and interpreters?

It eliminates misunderstandings, aids in the detection checks from compilers, and ensures programs are in line with the language guidelines.

Identify the terminals and non-terminals in the following rule:

```
<program> ::= <statement> {<statement>}  
<statement> ::= "if" "(" <condition> ")" "{" <body> "}" ["else" "{" <body> "}"]  
                | "while" "(" <condition> ")" "{" <body> "}"  
                | "print" "(" <expression> ")"
```

4.

Non-terminals: `<program>`, `<statement>`, `<condition>`, `<body>`, `<expression>`

Terminals: `"if"`, `"("`, `)"`, `"{"`, `"}"`, `"else"`, `"while"`, `"print"`

