Grammar (EBNF)

This document provides the grammar design for the BolBachchan programming language using Extended Backus-Naur Form (EBNF). The language syntax is inspired by Hindi-English hybrid (Hinglish) keywords, and it supports core programming constructs.

```
program         = { statement } ;

statement       = declaration
                | assignment
                | print
                | ifStatement
                | whileLoop
                | forLoop
                | expression ";" ;

declaration     = datatype identifier ";" ;
assignment      = "rakho" identifier "=" expression ";" ;
datatype        = "int" | "bool" | "string" ;

print           = "bolBhai" "(" expression ")" ";" ;

ifStatement     = "agar" "(" expression ")" "toh" "{" { statement } "}"
                [ "nahiToh" "{" { statement } "}" ] ;

whileLoop       = "jabTak" "(" expression ")" "{" { statement } "}" ;

forLoop         = "baarBaar" "(" assignment expression ";" assignment ")"
                "{" { statement } "}" ;

expression      = ternary | logical_expr ;

ternary         = logical_expr "?" expression ":" expression ;

logical_expr    = relational_expr { logical_op relational_expr } ;
relational_expr = arith_expr [ relationalOp arith_expr ] ;

arith_expr      = term { ("jodo" | "ghatao") term } ;
term            = factor { ("guna" | "bhaag") factor } ;

factor          = number
                | string
                | boolean_op
                | identifier
                | identifier increment_op
                | "(" expression ")" ;
```

```
relationalOp    = "badaHai" | "chhotaHai" | "barabarHai" ;
logical_op      = "&" | "|" ;
boolean_op      = "true" | "false" ;
increment_op    = "++" | "--" ;
identifier      = letter { letter | digit } ;
number          = digit { digit } ;
string          = "" { character } "" ;
letter          = 'a'..'z' | 'A'..'Z' ;
digit           = '0'..'9' ;
character       = letter | digit | ' ' | ',' | '.' ;
```