Assignment #02

Compiler Construction

Submitted by: Sardar M. Suleman
L1F22BSCS0255
G 5

Task #01 (Language Overview)

(Purpose)

=> Pakwaan++ is a cooking based programing language. Main idea of this language is to connect programming with daily life cooking so beginner can easily understand the logic like following a recipe. Programming statements work like cooking steps such as preparing ingredients, checking conditions and serving food.

(syntax style)

→ The syntax style of Pakwaan++ is similar to C++ but cooking related. Urdu keywords are used to make the language easy and intresting

(Reason for choosing your keywords)

→ These Keywords are choosen because they are meaningfull and easy to remember. When students read the code it feel like reading cooking instruction

From Phase 1

5 Keywords

Pakaai Shuru
Agar Halat
Jabtak Ghoom
Pura Hissa
Dawat Pesh

3 operators

+ (Add)
- (Remove)
/ (Divide)

Punctuation

{} (Block start/end)
; (intruction end)
()

Task #02 (Grammar Definition) (CFG)

(Non - terminals)

<Program> , < RecipeBlock> , < steplist> , < step>
, <Decl> , < Assign> , < ifcond> , < loop> , < output> , < Expr>

Task #03 (Sample Production Rule)

<Program> → Pakaai shuru( ) < RecipeBlock>
< RecipeBlock> → { < Steplist> }
< StepList> → < Step> < Steplist> | ϵ
< Step> → < Decl> | < Assign> | < ifcond> | < loop> | < output>
< Decl> → Pusa Hissa identifier ;
< Assign> → identifier = < Expr> ;
< Ifcond> → Agr Halat (< Expr>) < RecipeBlock>
< loop> → Jabtak Ghoom (< Expr>) < RecipeBlock>
< output> → Dawat Pesh identifier ;
< Expr> → identifier | integer | identifier + identifier

Task # 04 (First and follow sets)

(1) 1st Non-Terminal = <Step>

<Step> -> <Decl> | <Assign> | <IfCond> | <Loop> | <Output>

First (<Step>) =

{ Purattissa, identifier, Agar Halat, Jabtalc Ghoom, Dawat Pesh }

=> each product of step starts with different keyword that why its is in first set.

(2) 2nd Non-terminal = <Steplist>

First (<Steplist>) =

{ Purattissa, identifier, Agar Halat, Jabtalc Ghoom, Dawat pesh, ε }

(Production)

<StepList> -> <step> <steplist> | ε

Follow (<steplist>) = { }

=> <steplist> is used inside <RecipeBlock> -> } <steplist> }
so after <Steplist>, the closing brace } can appear.
which is included in the follow set

Task #05 (Ambiguity check)

Q Is the grammer Abiguous

NO, the grammer is not ambiguous

⇒ Because every statement starts with a unique keyword or identifier such as PuraHissa So there is no confusing in selecting production rule so the grammer is clear and deterministic

Task # 06 (Parse Tree Construction)

Program

```
Pakaai Shuru() {
    pura Hissa birgami;
    birgami = 5;
    Agar Halat (birgami) {
        Dawat Pesh birgami;
    }
}
```

(Parse Tree)

Task # 07 (Error Scenarios)

① Error Snippet :-

Pura Hissa chahiye

→ Line 2

→ Rule violated → Decl → Pura Hissa identifier ;

→ Expected token → ; at end

② Error Snippet

Agar Halat chahiye) {

Line → 2

Rule violated → <IfCond> → Agar Halat (<Exps>) <Recipe Block>

Expected token → (