University of Guelma

Computer Science Department **3rd year Computer Science Engineer AI**

Module: Compilation **Academic year: 2025/2026**

Series of Practical Work

We have a grammar of a programming language (**MiniPascal-fr** or **MP-FR**) in Backus Normal Formal-BNF-, you are asked to implement the **Scanner and Parser** of the MP-FR compiler.

1. Backus Normal Form:

	Representation	Explanation
Rule	::=	Sometimes = is used
Non-terminal	<….>	We use both comparison symbols < and > (<Non-Terminal>).
Terminals	**debut**	In lowercase
Concatenation	**ab**	Represents the sequence ab.
Choice	**a/b**	Either a, or b.
Option	**[a]**	Nothing or one occurrence of **a**.
Repetition	**[a]* or {a}**	Any number of **a's** including none.

2. Required tasks:

1. Provide the design and implementation of a scanner (**Scanner2025**) for the **MP-FR language.** The result of the scanner will be stored in a file called **TOKEN**, which the parser will use. **You must also display on screen the sequence of tokens, the contents of the symbols table, and any errors with their types**.

2. Provide the design and implementation of a recursive descent parser (**Parser2025)** for the **MP-FR** language. The parser's output will be stored in a file called **RULE**. **You must also display on screen the sequence of rules corresponding to the source program and any errors with their types**.

3. Definition of the grammar of the MiniPascal-Fr language.

<ProgrammePascal>::= **programme** <NomProgramme> **;** <Corps> **.**

<Corps>::=[<PartieDéfinitionConstante>][<PartieDéfinitionVariable>] <InstrComp>

<PartieDéfinitionConstante>::=**constante**<DéfinitionConstante>{<DéfinitionConstante>}

<DéfinitionConstante> ::=<NomConstante>**=**<Constante> **;**

<PartieDéfinitionVariable>::= **variable**<DéfinitionVariable>{<DéfinitionVariable>}

<DéfinitionVariable> ::=<GroupeVariable> **;**

<GroupeVariable> ::=<NomVariable>{**,** <NomVariable>} **:** <NomType>

<NomVariable> ::=<Nom>

<Nom> ::= <lettre><chiffre>{<lettre>/<chiffre>}

<Lettre> ::= ‘’**A**’’/ ‘’**B**’’/…./ ‘’**Z**’’/’’**a**’’/…/’’**z**’’

<Chiffre> ::=**0**/**1**/…./**9**

<NomType> ::=**entier/ reel**

<NomConstante>::=<Nom>

<Constante>::= <Nombre>/<NomConstante>

<NomProgramme>::= <Nom>

<Nombre> ::=<Chiffre> {<Chiffre>}/<Chiffre><Chiffre> {<Chiffre>}**.**<Chiffre>{<Chiffre>}

<InstrComp> ::=**debut** <Instruction>{ **;** <Instruction>} **fin**

<Instruction> ::= <InstructionAffectation> / <InstructionSi> / <InstructionTantque>/
<InstructionRépéter>/<InstrComp> /<InstructionPour> /<Vide>

<InstructionAffectation> ::=<NomVariable> **:=** <Expression>

<Expression> ::= <ExpressionSimple>[<OperateurRelationnel><ExpressionSimple>]

<OperateurRelationnel> ::= **<**/ **>**/ **=**/ **<=**/ **>=**/ **<>**

<ExpressionSimple> ::= [<OperateurSigne>]<Terme>{<OperateurAddition><Terme>}

<OperateurSigne> ::= **+**/**-**

<OperateurAddition> ::= +/**-**/**ou**

<Terme> ::=<Facteur>{<OperateurMult><Facteur>}

<OperateurMult> ::=***** / **div** / **mod** / **et**

<Facteur> ::=<Constante>/<NomVariable>/**(**<Expression>**)**

<InstructionSi> ::=**si** <Expression> **alors** <Instruction> [**sinon** <Instruction>]

<InstructionTantque> ::=**tantque** <Expression> **faire** <Instruction>

<InstructionRépéter> ::= **repeter** <Instruction> **jusqu’a** <Expression>

<InstructionPour> ::= **pour** <NomVariable> **allant de** <Constante> **a** <Constante> [**pas**
<Constante>] **faire** <Instruction>

<Vide> ::= ‘’

Remarks:

- The deadline for validating **Scanner2025** is fixed **on the 1st of December 2025**.
- The deadline for validating **Parser2025** is fixed **on the 12th of January 2026**.
- The responsible teacher of the practical work will validate the student's programs. So, the students have to follow his instructions.
- Students can be grouped into pairs (but not into trios).