**TRABALHO nO2: especificação da sintaxeda linguagem 2020.2**

**INSTRUÇÕES:**

1. O trabalho deve ser desenvolvido pela mesma equipe do desenvolvimento do compilador.
2. A especificação do TRABALHO nº2 estará disponível no AVA3, na aba COMPILADOR, arquivo trabalho no2.doc, a partir das **22h30 do dia 21 de setembro**.
3. Coloque o nome dos integrantes da equipe e as respostas das questões nos campos apropriados.
4. Poste o arquivo com as respostas na **pasta** da sua equipe no AVA3, na aba COMPILADOR, até às **9h do dia 28 de setembro**.

**IDENTIFICAÇÃO:**

|  |
| --- |
| **Jadiel dos Santos**  **Gian Carlo Giovanella**  **Matheus Mahnke** |

Construa **UMA ÚNICA** gramática, usando a notação BNF, que defina as regras sintáticas para escrever um programa na linguagem 2020.2, conforme as descrições abaixo. Considere que:

1. os símbolos terminais são os *tokens* especificados no TRABALHO no1, além das palavras reservadas e dos símbolos especiais usados nas especificações (forma geral) apresentadas abaixo;
2. uma vez descrita a forma geral de uma estrutura sintática, a mesma “vale” para toda a linguagem, por exemplo, na forma geral de um programa foi descrita a <lista de comandos>, sendo essa descrição válida para o programa, os comandos de seleção e de repetição;
3. as regras sintáticas para o não-terminal <expressão> (expressões aritméticas, lógicas e relacionais) serão especificadas posteriormente.

forma geral de um programa



onde:

* <lista de comandos> deve conter uma ou mais ocorrências de <comando>s. Os <comando>s podem ser de atribuição, de entrada de dados, de saída de dados, de seleção ou de repetição, conforme especificado abaixo

forma geral do comando de atribuição

identificador **=** <expressão>

ou identificador **+=** <expressão>

ou identificador **-=** <expressão>

forma geral do comando de entrada de dados

identificador **=** **input (** cte\_str **)**

ou identificador **=** <tipo> **(** **input (** cte\_str **) )**

onde:

* cte\_str é opcional.
* <tipo> pode ser int ou float ou str

forma geral dos comandos de saída de dados

**print (** <lista de expressões> **)**

onde:

* em <lista de expressões> deve conter uma ocorrência de <expressão>. Caso existam mais ocorrências, as mesmas são separadas uns das outras por vírgula ( , )

forma geral do comando de seleção

**if** <expressão> : <lista de comandos>

**elif** <expressão> : <lista de comandos>

**else :** <lista de comandos> **end**

onde:

* a cláusula **elif** <expressão> : <lista de comandos> pode ocorrer zero ou mais vezes
* a cláusula **else :** <lista de comandos> é opcional

forma geral do comando de repetição

**while** <expressão> **:** <lista de comandos> **end**

ou **for** identificador **in range** **(** <expressão> **) :** <lista de comandos> **end**

ou **for** identificador **in** identificador **:** <lista de comandos> **end**

**colocar aqui a gramática usando notação BNF**

|  |
| --- |
| **gramática** |
| <programa> ::= <lista\_comandos>  <lista\_comandos> ::= <comando> | <comando> <lista\_comandos>  <comando> ::= <atribuicao> | <entrada\_dados> | <saida\_dados> | <selecao> | <repeticao>  <atribuicao> ::= identificador **=** <expressao> | identificador **+=** <expressao> | identificador **-=** <expressao>  <entrada\_dados> ::= identificador **=** **input** **(** <entrada> **)** | identificador = <tipo> **(** **input** **(** <entrada> **)** **)**  <entrada> ::= **cte\_str** | î  <tipo> ::= **int** | **float** | **str**  <saida\_dados> ::= **print** **(** <lista\_expressoes> **)**  <lista\_expressoes> ::= <expressao> | <expressao> **,** <lista\_expressoes>  <selecao> ::= **if** <expressao> **:** <lista\_comandos> <senao\_se> <senao> **end**  <senao\_se> ::= **elif** <expressao> **:** <lista\_comandos> <senao\_se> | î  <senao> ::= **else** **:** <lista\_comandos> | î  <repeticao> ::= **while** <expressao> **:** <lista\_comandos> **end** | **for** identificador **in** **range** **(** <expressao> **)** **:** <lista\_comandos> **end** | **for** identificador **in** identificador **:** <lista\_comandos> **end** |