**TRABALHO nº1 – especificação dos tokens da linguagem 2020.1**

**EQUIPE (a mesma do desenvolvimento do compilador):**

Para as linguagens regulares (dos *tokens*) denotadas abaixo, especifique **definições regulares**:

| **DEFINIÇÕES REGULARES AUXILIARES** (válidas para todas as especificações abaixo) |
| --- |
| **letra = [A-Z] | [a-z]**  **dig = [0-9]**  **dig1 = [1-9]** |

|  |
| --- |
| **identificador** |
| 1. **(2,0 pontos) identificador**:é qualquer palavra composta por letras (maiúsculas ou minúsculas), dígitos e *underline* ( \_ ). Um identificador deve começar com uma letra. Na sequência podem existir outras ocorrências (zero ou mais) de letras ou de dígitos ou de *underlines*, desde que não possua *underlines* consecutivos.   **DEFINIÇÃO REGULAR:**  **identificador = letra (letra | dig )\* (\_ (letra | dig)+ )\* (\_)?** |
| **constante** inteira |
| 1. **(1,2 pontos)** **constante** inteira:é qualquer palavra composta por no mínimo um dígito decimal (0 a 9), que não possui 0 desnecessários à esquerda. Observa-se que a constante 0 é válida. No entanto, são exemplos de constantes inválidas: 00, 01, 019, 010, 001, 0019   **DEFINIÇÃO REGULAR:**  **cInteira = dig1 dig\* | dig** |
| **constante** real |
| 1. **(1,6 pontos)** **constante** real:é qualquer palavra composta por uma parte inteira, concatenada com uma parte fracionária, nessa ordem. A parte inteira é composta por no mínimo um dígito decimal, que não possui 0 desnecessários à esquerda. A parte fracionária é composta por um ponto decimal ( **\.** ), seguido por no mínimo um dígito decimal, que não possui 0 desnecessários à direita. Observa-se que a constante 0.0 é válida. No entanto, são exemplos de constantes inválidas: 00.1, 1.10, 12.030, 01230.045, 12345.12340   **DEFINIÇÃO REGULAR:**  **cReal = (dig1 dig\* | dig) \. (dig\* dig1 | dig)** |
| **constante** binária |
| 1. **(1,0 ponto)** **constante** binária: é qualquer palavra que inicia com um *hashtag*, seguido pela letra **b** (maiúscula ou minúscula) ( #b ou #B ), seguida por mínimo um dígito binário (0 ou 1).   **DEFINIÇÃO REGULAR:**  **cBinaria = # (b | B) (0 | 1)+** |
| **constante** hexadecimal |
| 1. **(1,0 ponto)** **constante** hexadecimal: é qualquer palavra que inicia com um *hashtag*, seguido pela letra **x** (maiúscula ou minúscula) ( #x ou #X ), seguida por no mínimo um dígito hexadecimal (0 a 9, a, b, c, d, e, f).   **DEFINIÇÃO REGULAR:**  **cHexadecimal = # (x | X) (dig | [a-f])+** |
| **constante** string |
| 1. **(1,2 pontos)** **constante** string: é qualquer palavra delimitada por aspas duplas ( **\"** ), uma no início e outra no final. Uma constante string pode possuir entre as aspas duplas zero ou mais ocorrências de quaisquer caracteres, exceto aspa dupla ( **\"** ), quebra de linha ( **\n** ) e barra invertida ( **\\** ).   **DEFINIÇÃO REGULAR**:  **cString = \“ [^ \” \n \\]\* \”** |
| **comentário de linha** |
| 1. **(1,0 ponto)** **comentário de linha**: a equipe deve definir o comentário de linha, apresentando a descrição textual e a definição regular correspondente. Observa-se que o comentário de linha não deve começar com letra, dígito, *hashtag*, aspas duplas ou barra invertida.   **DESCRIÇÃO TEXTUAL**: inicia com // seguidas por zero ou mais ocorrências de quaisquer caracteres, exceto quebra de linha ( \n ).  **DEFINIÇÃO REGULAR**:  **comentarioLinha = // [^\n]\*** |
| **comentário de bloco** |
| 1. **(1,0 ponto)** **comentário de bloco**: a equipe deve definir o comentário de bloco, apresentando a descrição textual e a definição regular correspondente. Observa-se que o comentário de bloco não deve começar com letra, dígito, *hashtag*, aspas duplas ou barra invertida.   **DESCRIÇÃO TEXTUAL**: inicia com /\*, seguido por zero ou mais ocorrências de quaisquer caracteres, exceto \*/ nessa ordem, e termina com \*/.  **DEFINIÇÃO REGULAR**:  **comentarioBloco = / \\* ( [^\\*] | (\\*)+[^/\\*] )\* (\\*)+ /** |