|  |  |  |
| --- | --- | --- |
| FACENS_NovoLogo | **Compiladores – Laboratório** | |
| **Definição de Linguagem de Programação** | |
| Nome: Anderson Martins da Silva | | RA: 140513 |
| Nome: Marco Aurélio Daniel | | RA: 141208 |

|  |
| --- |
| **1. Descrição da Linguagem** |
| Corcunrex é uma linguagem case sensitive baseada em blocos de desenvolvimento, possuindo um bloco responsável pela declaração de todas as variáveis, outro bloco responsável pela inicialização das variáveis e o último bloco onde se encontra todo o código a ser desenvolvido. A linguagem possui uma estrutura de decisão e uma de repetição, possui ainda operações aritméticas básicas, funções de entrada e saída e operações lógicas. |

|  |  |
| --- | --- |
| **2. Símbolos Terminais Fixos** | |
| Composição dos símbolos terminais da gramática que representa uma linguagem que não se alteram. | |
| Símbolo | Significado |
| variables | Símbolo inicial que representa a declaração das variáveis. |
| init | Símbolo que representa a inicialização das variáveis. |
| code | Símbolo que representa onde ficará o código. |
| ( | Símbolo que representa o início das operações lógicas em uma estrutura de decisão, inicio de bloco para receber variáveis nas funções de entrada de saída, inicio de bloco onde contém as variáveis em uma operação aritmética, inicio de bloco de operações lógicas em uma estrutura de repetição, inicio de bloco para variáveis em uma operação lógica. |
| ) | Símbolo que representa o fim das operações lógicas em uma estrutura de decisão, fim de bloco para receber variáveis nas funções de entrada de saída, fim de bloco onde contém as variáveis em uma operação aritmética, fim de bloco de operações lógicas em uma estrutura de repetição, fim de bloco para variáveis em uma operação lógica. |
| { | Símbolo que representa o início dos blocos de desenvolvimento e dos nomes das variáveis na declaração de variáveis. |
| } | Símbolo que representa o fim dos blocos de desenvolvimento e dos nomes das variáveis na declaração de variáveis. |
| [ | Símbolo que representa o início de um bloco de código. |
| ] | Símbolo que representa o fim de um bloco de código. |
| & | Operador lógico que verifica se todos os valores são verdades. |
| | | Operador lógico que verifica se pelo menos um valor é verdade. |
| > | Operador lógico que compara se um valor é maior que outro. |
| < | Operador lógico que compara se um valor é menor que outro. |
| = | Operador lógico que compara se um valor é igual a outro. |
| + | Símbolo que representa uma operação de soma. |
| - | Símbolo que representa uma operação de subtração. |
| \* | Símbolo que representa uma operação de multiplicação. |
| / | Símbolo que representa uma operação de divisão. |
| In | Símbolo que representa uma função de entrada de dados. |
| out | Símbolo que representa uma função de saída de dados. |
| loop | Símbolo que representa uma estrutura de repetição. |
| ? | Símbolo que representa uma estrutura de decisão. |
| : | Símbolo que representa uma atribuição de valor em uma variável. |

|  |  |
| --- | --- |
| **3. Símbolos Terminais Variáveis** | |
| Composição dos símbolos terminais da gramática que representa uma linguagem que podem ser alterados. | |
| Símbolo | Significado |
| var | Composição de letras e do caractere \_. Regex: [a-z]|\_ |
| intI | Composição de números. Regex: [0-9] |
| IntF | Composição de números com casas decimais separados por . Regex: idI|. |

|  |  |
| --- | --- |
| **4. Principais Operadores** | |
| Definir os principais operadores que podem ser aplicados entre variáveis, valores e constantes | |
| Símbolo | Significado |
| + | Soma dois números |
| \* | Multiplica dois números |
| / | Divide dois números |
| - | Subtrai dois números |

|  |  |
| --- | --- |
| **5. Principais Tipos de Dados** | |
| Definir os principais tipos de dados da sua linguagem | |
| Símbolo | Significado |
| int | Armazena um número inteiro de 32 bits |
| float | Armazena um número com notação de ponto flutuante simples |

|  |  |
| --- | --- |
| **6. Principais Instruções** | |
| Definir as principais instruções presentes na linguagem, inclusive condicionais, loops, declaração de variáveis, etc | |
| Símbolo | Significado |
| variável: valor | Atribuição de valor em uma variável |
| int{variavel1 variavel2 variavelN} | Declarar uma ou mais variáveis do tipo inteiro de 32 bits |
| float{variavel1 variavel2 variavelN} | Declarar uma ou mais variáveis do tipo ponto flutuante simples |
| +(variavel1 variavel2 variavelN) | Soma duas ou mais variáveis |
| -(variavel1 variavel2 variavelN) | Subtrai duas ou mais variáveis |
| \*(variavel1 variavel2 variavelN) | Multiplica duas ou mais variáveis |
| /(variavel1 variavel2 variavelN) | Divide duas ou mais variáveis da esquerda para a direita |
| in(variavel1 variavel2 variavelN) | Receber valores do usuário em um ou mais variáveis |
| out(variavel1 variavel2 variavelN) | Mostrar valores na tela de uma ou mais variáveis |
| ?(/\*Operação lógica\*/)[/\*Código\*/][/\*Código\*/] | Estrutura de decisão com o primeiro bloco de código para condição verdadeira e o segundo caso a condição seja falsa. |
| ?(/\*Operação lógica\*/)[/\*Código\*/] | Estrutura de decisão com o primeiro bloco de código para condição verdadeira. |
| loop(/\*Operação lógica\*/)[/\*Código\*/] | Estrutura de repetição com bloco de código para condição verdadeira. |
| VARIABLES{/\*Código\*/} | Bloco para declaração de variáveis. |
| INIT{/\*Código\*/} | Bloco para inicialização das variáveis. |
| CODE{/\*Código\*/} | Bloco para lógica de implementação. |
| &(variavel1 variavel2 variavelN) | Operador lógico que verifica se todos os valores são verdades entre uma ou mais variáveis. |
| |(variavel1 variavel2 variavelN) | Operador lógico que verifica se alguma variável é verdadeira entre uma ou mais variáveis. |
| >(variavel1 variavel2) | Operador lógico que compara se um valor é maior que outro. |
| <(variavel1 variavel2) | Operador lógico que compara se um valor é menor que outro. |
| =(variavel1 variavel2 variavelN) | Operador lógico que compara se uma variável é igual a outra. |

|  |  |
| --- | --- |
| **7. Exemplos** | |
| Apresentar exemplos de códigos feitos em sua linguagem e o análogo em C | |
| Minha Linguagem | C |
|  |  |
|  |  |

|  |
| --- |
| **8. Gramática** |
| G = ({A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, X}, {variables, init, code, int, float, var, +, -, \*, /, in, out, ?, [, ], (, ), {, }, loop, &, |, >, <}, Regras, S)  Regras = {  S -> variables{A}init{B}code{C}  A -> D{E} A|ε  D -> int|float  E -> var  B -> E: F B|ε  F -> D|E|G  C -> B C|G C|H C|I C|J C|K C|ε  G -> L(M)  L -> +|-|\*|/  M -> F F M|F M|ε  H -> N(M)  N -> in|out  I -> ?(K)O  K -> Q|R  Q -> X(M)  X -> &||  R -> T(F F)  T -> >|<  O -> P|PP  P -> [C]  J -> loop(K)P  } |