Dupla:
Carlos Eduardo Xavier
Elygledson José da Silva Bezerra

1- ESTRUTURA GERAL DO PROGRAMA

FUNÇÃO PRINCIPAL

A nossa linguagem se inicializa declarando uma (function main() {}). O conteúdo inicial deve ser declarado dentro de chaves.

- **DECLARAÇÃO DE VARIÁVEIS**

Para qualquer variável ou constante que declararmos na nossa linguagem teremos que deixar claro o tipo logo após o nome seguido de dois pontos < : >.

- **DECLARAÇÃO FUNÇÕES**

Exemplo:

```
function NOME(<PARÂMETROS>): <TIPO DE RETORNO> {
    <CÓDIGO>
}
```

Inicialmente toda função precisa iniciar com o identificador 'function' logo após declaramos o nome dela seguido de dois < : > em seguida especificamos o tipo de retorno que pode ser ex .: int, char, string, bool, float… Os parâmetros devem ser tipados seguindo a mesma ordem como fazemos na declaração de variáveis separados por vírgula.
Todo código de uma função deve ser escrito entre chaves.

- **UTILIZAÇÃO DE PONTO E VÍRGULA**

Em todos os casos, menos em escopos condicionais, funcionais e durante o uso de laços de repetições.

2 - TIPOS DE DADOS E NOMES

NOME	TIPO
int	TIPO DE DADO
float	TIPO DE DADO
string	TIPO DE DADO
char	TIPO DE DADO
function	FUNÇÃO
return	INSTRUÇÃO DE RETORNO
void	TIPO DE DADO

if	CONDICIONAL
else if	CONDICIONAL
else	CONDICIONAL

Array	CONJUNTO DE DADOS
boolean	TIPO DE DADO
true	VALOR BOOLEANO
false	VALOR BOOLEANO
input	COMANDO DE ENTRADA
print	COMANDO DE SAÍDA
while	ESTRUTURA DE REPETIÇÃO
for	ESTRUTURA DE REPETIÇÃO
break	INSTRUÇÃO DE INTERRUPÇÃO
and	OPERADOR LÓGICO
not	OPERADOR LÓGICO
or	OPERADOR LÓGICO

3 - CONSTANTES LITERAIS

boolean	false
string	null
char	null
number	0
int	0
float	0.0

4 - OPERAÇÕES PARA CADA TIPO

int	Operações aritméticas, relacionais, atribuição, concatenação com string e chars
float	Operadores aritméticos, relacionais, atribuição, concatenação com string e char
char	Operador de atribuição, relacionais e concatenação
string	Operador de atribuição, relacionais e concatenação
boolean	Operador de atribuição, igualdade, desigualdade, lógicos e concatenação com string e char

5 - COERÇÃO SUPORTADAS

6 - CONJUNTOS DE OPERAÇÕES

6.1 OPERADORES ARITMÉTICOS

OPERADORES	DESCRIÇÃO
+	SOMA
%	RESTO DA DIVISÃO ENTRE
*	MULTIPLICAÇÃO
-	SUBTRAÇÃO

6.2 OPERADOR COM ATRIBUIÇÃO

OPERADOR	DESCRIÇÃO
=	ATRIBUIÇÃO DE VALOR

6.3 OPERADORES LÓGICOS

OPERADORES	OPERAÇÃO
and	CONJUNÇÃO
or	DISJUNÇÃO
not	NEGAÇÃO

6.4 OPERADORES RELACIONAIS

OPERADORES	OPERAÇÃO
!=	DIFERENÇA ENTRE DOIS OPERANDOS
==	IGUALDADE ENTRE DOIS OPERANDOS
>	MAIOR QUE
<	MENOR QUE
<=	MENOR IGUAL QUE
>=	MAIOR IGUAL QUE

6.5 CONCATENAÇÃO DE CADEIAS DE CARACTERES

Na linguagem de programação podemos concatenar tipo usando o operador ‘+’ entre strings e char ou string e char com int, float, number e boolean convertendo-os para o tipo string. Além disso, pode-se usar o concat() como citado no item 5.

6.6 ORDEM DE PRECEDÊNCIA E ASSOCIATIVIDADE

A precedência da linguagem é de cima para baixo, quanto mais ao topo os operadores estiverem maior a prioridade. No entanto, caso haja o uso de parênteses haverá um acréscimo de prioridade na expressão que está dentro dos parênteses.

PRECEDÊNCIA	OPERADORES	ASSOCIATIVIDADE
+	ADITIVO E CONCATENAÇÃO	ESQUERDA PARA DIREITA
-	ADITIVO	ESQUERDA PARA DIREITA
*/%	MULTIPLICATIVOS	ESQUERDA PARA DIREITA
> < >= <=	COMPARATIVOS	NÃO TEM
== !=	IGUALDADE	NÃO TEM
and or	CONJUNÇÃO	ESQUERDA PARA DIREITA
not	NEGAÇÃO	DIREITA PARA ESQUERDA

7 - Instruções

Nesta linguagem, após cada instrução declarada deve terminar pelo símbolo ‘;’ além disso todo bloco iniciado deve possuir chaves de abertura e encerramento.

Ex.:

```
function teste(): void {
   identificador: int =  0;
}
```

7.1 Atribuição

A atribuição nesta linguagem é feita com ajuda do sinal de igualdade “ = “ onde o lado esquerdo deve ter a variável, que deve ser declarada juntamente com seu tipo, e o lado direito obrigatoriamente um valor que corresponde ao tipo da variável.

8 - Estrutura condicional de uma via ou duas

A estrutura condicional ‘if’ representa um bloco de instruções em que caso a expressão lógica relacionada a ele for verdadeira o bloco será executado. Além disso, em seguida do if podemos utilizar a estrutura condicional else if que possui a mesma lógica do if. Por fim, temos a estrutura ‘else’ que só vai ser executada caso as estruturas condicionais que antecedem ele forem falsas.

Um ponto de observação em relação ao uso das estruturas, não pode ser utilizada o else if seguido de um else caso antes não tenha sido declarado um if.

Ordens aceitas:

9 - Estruturas interativas com controles lógicos

A estrutura de repetição while e for são suportadas pela nossa linguagem.

While - Podemos utilizar esta estrutura através de uma condição lógica que vai permitir a repetição do bloco enquanto for verdadeira. Para que o while funcione é necessário que ele seja seguido de parênteses com a condição lógica dentro, em seguida, vem o bloco de instruções que precisa está entre chaves e não precisa terminar com “;”. Além disso, caso seja necessário interromper o laço de repetição antes da condição lógica ser falsa deve-se utilizar a palavra reservada “break” seguido de “;” dentro do bloco.

For - Um pouco diferente do while, a estrutura de repetição for necessita de 3 parâmetros: o iterador(tipo int), a condição de pausa(expressão lógica) e o contador(tipo int) separador por vírgula. Para que o for funcione ele deve ser seguido de parênteses com obrigatoriamente os 3 parâmetros citados na respectiva ordem. Assim como o while, pode-se utilizar o break para encerrar antes da condição de pausa ser concluída.

10 - Dados de entrada e saída

Nesta linguagem, utilizamos dois comandos input() para entrada de dados e print() para saída.

10.1 Entrada

A nossa linguagem nos permite obter dados do usuário escritos no terminal através do comando input() onde dentro deste comando pode ser especificada uma mensagem. O retorno do input sempre será uma string, então deve-se converter o que foi lido para que seja compatível com o tipo da variável predefinido. Ver os métodos de coerção disponíveis na linguagem [capítulo].

10.2 Saída

Para exibirmos o que foi lido, definimos o comando print que pode receber como parâmetros quanto variáveis forem necessárias separadas por vírgula. Além disso, podemos escrever uma constante literal,uma string, entre aspas antes de passarmos os parâmetros dentro de parênteses. Usa-se “\n” caso necessitamos pular de linha pois o print exibe o conteúdo na mesma linha.

11 - Funções

Para iniciarmos uma função precisamos utilizar a palavra reservada ‘function’ e logo após escrever o nome da função seguido de parâmetros caso haja e depois declararmos o tipo do retorno da função. Em seguida, abrimos as chaves para iniciar o bloco de instruções onde devemos especificar o que a função deve fazer.