Especificação da Linguagem Simplificada Baseada em JavaScript

1. Introdução

Este documento apresenta a especificação formal da linguagem simplificada que será suportada pelo compilador a ser desenvolvido. A linguagem é baseada em JavaScript, mantendo sua sintaxe e semântica básicas, porém focada em um subconjunto simplificado para facilitar o desenvolvimento do compilador.

2. Tipos de Dados

A linguagem suporta os seguintes tipos básicos e derivados:

- number: Números inteiros e reais, sem distinção explícita entre os dois.
- string: Cadeias de caracteres delimitadas por aspas duplas ou simples.
- boolean: Valores lógicos true e false.
- null: Valor especial representando ausência de valor.
- undefined: Valor padrão para variáveis não inicializadas.
- array: Conjuntos indexados de elementos homogêneos ou heterogêneos.
- object: Coleção de pares chave-valor.

Exemplo:

```
let idade = 25;
const nome = "Lemuel";
let ativo = true;
let lista = [1, 2, 3];
let pessoa = {nome: "Ana", idade: 30};
```

3. Operadores e Precedência

A linguagem inclui os seguintes operadores, organizados pela precedência (do maior para o menor):

Precedência	Operadores	Descrição
1	()	Agrupamento
2	!	Negação lógica
3	*,/,%	Multiplicação, divisão e módulo
4	+,-	Adição e subtração
5	<,<=,>,>=	Comparações
6	==,!=,===,!==	Igualdade e diferença
7	&&	E lógico
8	||	Ou lógico
9	=	Atribuição

Exemplo:

let total = (5 + 3) * 2 > 10 && true;

4. Declaração de Variáveis e Escopo

- Variáveis locais são declaradas com let. Constantes, com const.
- O escopo é limitado ao bloco onde a variável é declarada.
- Não existe escopo global nesta linguagem simplificada.
- Variáveis devem ser declaradas antes do uso, não considerando hoisting.
- Constantes não podem ser reatribuídas.

Exemplo:

```
let contador = 0;
const PI = 3.14;

if (contador < 10) {
  let interno = contador * 2;  // variável apenas dentro deste bloco
}
```

5. Declaração de Funções

- Funções são declaradas usando function.
- Aceitam zero ou mais parâmetros.
- Valores podem ser retornados usando return.

Exemplo:

```
function soma(a, b) {
  return a + b;
}

let resultado = soma(3, 7);
```

6. Funções Nativas Suportadas

A linguagem suporta um conjunto básico de funções nativas:

- print(valor): Exibe o valor informado na saída padrão.
- parseInt(string): Converte uma string para número inteiro.
- parseFloat(string): Converte uma string para número real.
- typeof(valor): Retorna o tipo do valor.

Exemplo:

```
print("Teste");
let num = parseInt("123");
print(typeof(num));  // "number"
```

7. Sentenças de Controle e Atribuição

- Atribuições são feitas com =.
- Estruturas suportadas: if/else, while, for.
- if: executa bloco condicionalmente.
- while: executa enquanto a condição for verdadeira.
- for: loop tradicional com inicialização, condição e incremento.

Exemplo:

```
let i = 0;
while (i < 5) {
  print(i);
  i = i + 1;
}

for (let j = 0; j < 3; j = j + 1) {
  print(j);
}

if (i === 5) {
  print("Fim da contagem");
} else {
  print("Contagem incompleta");
}
```

Considerações Finais

Esta especificação servirá como base para construção do compilador, que analisará código fonte escrito nesse subconjunto simplificado de JavaScript, garantindo sintaxe e semântica definidas acima.