A expressão parseFloat(url.searchParams.get('num1')); no código faz duas coisas principais:

**Obtém o valor do parâmetro 'num1' da URL**:

- url.searchParams.get('num1') procura o valor do parâmetro num1 na URL.
- Por exemplo, se a URL for http://localhost:3000/?num1=5&num2=3, essa linha encontrará o valor "5" para num1 como uma *string*.

**Converte a string para um número decimal**:

- parseFloat() pega a string encontrada e a converte para um número decimal (ou número de ponto flutuante).
- Por exemplo, parseFloat("5") resulta em 5 como um número.

Portanto, parseFloat(url.searchParams.get('num1')); significa: "pegue o valor do parâmetro num1 na URL e o transforme em um número".

soma.js > server.listen() callback

```
// Importa o módulo 'http' do Node.js, que permite criar um servidor HTTP.
const http = require('http');

// Cria um servidor HTTP que escuta requisições na porta 3000.
const server = http.createServer((req, res) => {
  // Extrai a URL e o método da requisição.
  const url = req.url; // A URL requisitada.
  const method = req.method; // O método HTTP (GET ou POST).

  // Verifica se a URL começa com '/', que é o endpoint que vamos usar.
  if (url.startsWith('/')) {
    // Obtém os parâmetros da URL (query parameters).
    const queryParams = new URLSearchParams(url.split('?')[1]);
    const num1 = parseFloat(queryParams.get('num1')); // Obtém o valor do parâmetro 'num1'.
    const num2 = parseFloat(queryParams.get('num2')); // Obtém o valor do parâmetro 'num2'.

    // Verifica se ambos os números são válidos
    if (isNaN(num1) || isNaN(num2)) {
      res.setHeader('Content-Type', 'text/plain; charset=utf-8'); // Define o cabeçalho como texto simples com codificação UTF-8
      res.statusCode = 400; // Código de erro 400 (solicitação inválida)
      res.end('Por favor, forneça dois números válidos nos parâmetros "num1" e "num2".');
    } else {
      // Calcula a soma
      const soma = num1 + num2;

      // Prepara a resposta com a soma
      res.setHeader('Content-Type', 'text/plain; charset=utf-8'); // Define o cabeçalho como texto simples com codificação UTF-8
      res.statusCode = 200; // Código de sucesso 200 (OK)
      res.end(`A soma é: ${soma}`); // Envia a soma como resposta
    }
  } else {
    // Se a URL não for válida, retorna um erro 404 (Não Encontrado).
    res.setHeader('Content-Type', 'text/plain; charset=utf-8'); // Define o cabeçalho como texto simples com codificação UTF-8
    res.statusCode = 404; // URL não encontrada
    res.end('Rota não encontrada');
  }
});

// O servidor escuta na porta 3000.
server.listen(3030, () => {
  console.log('Servidor rodando na porta 3000...');
});

```

A expressão isNaN(num1) || isNaN(num2) está verificando se algum dos dois valores, num1 ou num2, **não é um número**.

Vamos dividir em partes para entender melhor:

**1.isNaN**: Esse é um comando que significa "é um valor que **não é número**?". Ele devolve true se o valor não for um número e false se for um número.

**2.isNaN(num1)**: Aqui, o código está perguntando: **"num1 não é um número?"**. Se num1 não for um número (como uma letra, por exemplo), isso vai ser true.

**3.||**: Esse símbolo significa "ou". Quer dizer que o código vai dar true se **qualquer uma das perguntas for verdadeira**.

**4.isNaN(num1) || isNaN(num2)**: No final, essa linha está perguntando:

- "num1 não é um número?"
- "OU num2 não é um número?"

Se qualquer uma dessas respostas for true (ou seja, se um deles não for um número), o resultado final será true.