**1. Controle de fluxo (original e intuitivo):**

* decida — Em vez de if, indicando uma decisão.
* caso\_contrário — Substituto para else, como "caso contrário".
* repita — Para laços de repetição, substituindo for ou while.
* pule — Ignorar a iteração atual do laço (substituto para continue).
* pare — Para sair de um laço ou função (em vez de break).

**2. Tipos de dados (descritivos e únicos):**

* número — Para números inteiros e decimais (em vez de int ou float).
* texto — Substituto para string, para representar cadeias de texto.
* sinal — Para valores booleanos (em vez de bool), sugerindo uma bandeira de verdadeiro ou falso.
* byte — Para dados de um caractere ou um número pequeno (em vez de char).
* vazio — Para uma variável ou função sem retorno (substituto de void).

**3. Declaração de variáveis e funções (inovadoras e diretas):**

* defina — Para declarar uma variável (substituto para var ou let).
* fixo — Para uma variável imutável, equivalente a const.
* faça — Para declarar uma função (substituindo func ou def).
* crie — Para instanciar um novo objeto ou classe (em vez de class ou new).

**4. Estruturas de dados (originais e com propósito):**

* pacote — Substituto de array, representando um conjunto de valores.
* mapa\_chave — Para uma estrutura de pares chave-valor (em vez de map ou dict).
* conjunto — Para uma coleção de dados múltiplos (substituindo struct ou objeto).

**5. Operadores lógicos e comparações:**

* ambos — Para o operador lógico and.
* um\_ou\_outro — Para o operador lógico or.
* nenhum — Para o operador lógico not.
* é — Para comparação de igualdade (em vez de ==).
* não\_é — Para comparação de desigualdade (em vez de !=).

**6. Fluxo de exceção (com estilo próprio):**

* tente — Para tentar executar um bloco de código (em vez de try).
* resgate — Para capturar uma exceção (em vez de catch).
* falhe — Para lançar uma exceção (substituindo throw).

**7. Memória e alocação (caso precise de controle):**

* alocar — Para alocar memória dinamicamente (em vez de malloc).
* libere — Para liberar memória alocada (substituto de free).

**8. Outros comandos especiais (criativos e originais):**

* use — Para importar módulos ou bibliotecas (em vez de import).
* ligue — Para conectar ou incluir pacotes ou módulos (como include ou require).
* fluxo — Para iniciar um bloco de código (equivalente a { ou begin).
* retorne — Para finalizar a execução de uma função e devolver um valor (em vez de return).