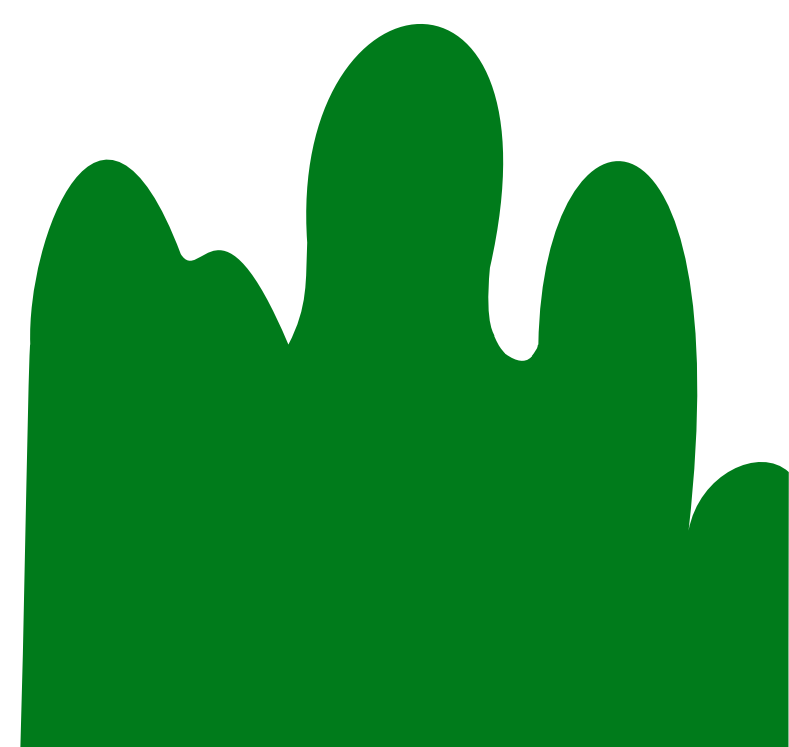
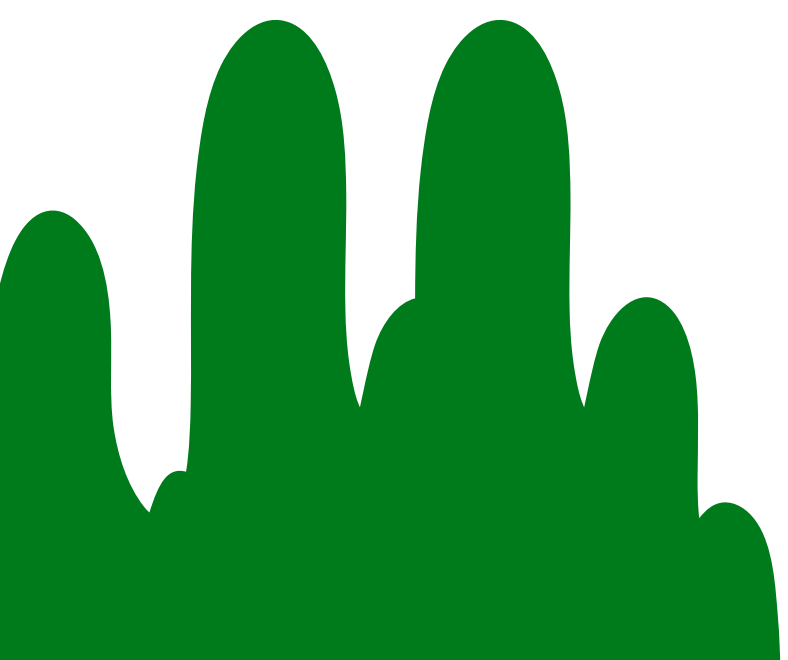




LINGUAGEM GLURR'IK E COCOONVM

Desenvolvido por Cynthia Naoko Yasutake



MOTIVAÇÃO

- A Glurr'ik surgiu da curiosidade de um investigador gosma da Colmeia, que desejava analisar o comportamento dos casulos e o fluxo da mente coletiva sem expor seres vivos a perigos;
- Simula processos biológicos de transformação;
- Cria uma ponte de biocomputação e sistemas digitais;
- Conceito de linguagem viva;
- Projeto inspirado na minha história fictícia da sociedade gosma.

CARACTERÍSTICAS

- Sintaxe: Inspirada em sons e palavras do idioma gosma (ex: Srrl, Vleth, Frral)
- Paradigma: Estruturada e biológica (mistura de C + ritual simbólico)
- Variáveis: Vleth — criação de matéria gosma (declaração)
- Condicional: Frral / Shrelk — estruturas de decisão
- Loop: Zrran — iteração biológica (ciclo de energia)
- Retorno: Srryl — devolve o estado final do casulo
- Literal especial: Eshl — essência primordial das gosmas
- Comentários: Começam com # ou //
- Saída: Gera código Assembly interpretado pela CocoonVM

CURIOSIDADES

- Os monstros de gosma foram criados após a explosão da Pedra Sombria.
- A Colmeia funciona como uma mente coletiva universal.
- A CocoonVM simula o processo de encasulamento, com registradores que representam:
 - R1: Massa (vitalidade da gosma)
 - R2: Energia Sombria
 - R3: Mutação
 - R4: Fluxo coletivo (vínculo com a Colmeia)
- Cada instrução da VM é uma ação biológica:
- MUTAR, CONDENSAR, ABSORVER, PARTILHAR...

EXEMPLOS

```
Srrl riralu() {  
  Vleth sama Eshl  
  Hrrash sama -> mava  
  Drazh sama, mava Eshl gaga  
  Frral (100) {  
    Zrran (10) { Srryl 0 }  
  } Shrelk {  
    Srryl 1  
  }  
}
```

Trecho de código (alto nível)

```
/* literal Eshl */  
// MUTATE sama ... expr below  
MUTAR sama  
PARTILHAR sama, mava  
CONDENSAR sama, mava -> gaga  
/* push num 100 */  
/* push num 10 */  
/* push num 0 */  
RETURN  
LABEL loop_start_0  
IFZERO loop_end_0  
JMP loop_start_0  
LABEL loop_end_0  
/* push num 1 */  
RETURN  
// IF start  
IFGT if_true_0  
JMP if_false_0  
LABEL if_true_0  
JMP if_end_0  
LABEL if_false_0  
LABEL if_end_0  
LABEL riralu  
LABEL riralu_end
```

Código assembly gerado

EXEMPLOS

```
yititlaaokoa50LakE@LAPTOP-IRANLUBC ./MITC/0/05  
[VM] finished. R1=0 R2=0 R3=2 R4=5  
[VM] history:  
  MUTAR sama  
  PARTILHAR  
  CONDENSAR
```

Saída da VM.

A função `riralu()` simboliza o ciclo de vida de um casulo gosma: formação, ligação, união, escolha e renascimento. Ela apresenta todas as estruturas fundamentais da língua Glurr'ik: variáveis, operações biológicas, condições (`Frral`) e laços (`Zrran`)

EXEMPLOS

A função riralu() simboliza o ciclo de vida de um casulo gosma: formação, ligação, união, escolha e renascimento.
Ela apresenta todas as estruturas fundamentais da língua Glurr'ik: variáveis, operações biológicas, condições (Frral) e laços (Zrran)

EXEMPLOS

- Srrl riralu() { : Declara o processo principal
- Vleth sama Eshl : Cria a massa inicial do casulo a partir da essência primordial ("Eshl")
- Hrrash sama -> mava : Conecta sama a mava, transferindo parte da energia
- Drazh sama, mava Eshl gaga : Funde sama e mava gerando uma nova forma gaga
- Frral (100) { : Inicia uma condição (como if (100)) — aqui sempre verdadeira, usada para mostrar o bloco de decisão
- Zrran (10) { Srryl 0 } : Executa um laço (loop) 10 vezes — simula pulsos ou ciclos de energia
- Shrelk { Srryl 1 } : Bloco "else" — executado se a condição falhasse (não é o caso)
- } : Fecha o processo