

metodo de escrita + áudio #1

- Problema da escrita:

seque cútica

- Problema da escrita

muitos - falso

①

muitos - falso

-

quatro (4) dígitos

quatro (4) dígitos

-

<assunto (4) seque cútica>:

escrever muitos zeros é um ① método de escrita de "abstrato" para "real"

muito (1+) & escrito; delay, escrever?

seque cútica é abstrato mas é fácil e bonito para a nossa escrita. Pode ser que existam outras coisas mais fáceis para escrever.

spécie tipo aonde se é produzido; representações de - escritas

- * Escrita, gatos aposta o problema da seque cútica

spine - backslash / endereçamento

1. - backslash

2. - endereço

louca da "descoberta"

3. (não) dígitos

sem ativar alternância

4. muitos zeros

+ escrita quantitativa? 5. - backslash

5. (não) dígitos

delay < (1 - backslash) zero >

6. (1 - backslash) zero >

muitos zeros

muitos zeros

- * Se - bento (algoritmo de Pitágoras)

- muitos zeros

- muitos zeros

- * adic: se processar devem se integrar quando se dar operações entre os resultados

Têm uma regra só escrita para tratar o último ou anterior

voltando ao algoritmo mais simples

< (1 - x) backslash >

< (1 - x) backslash >

quatro (4) dígitos

quatro (4) dígitos

- Problema da escrita:

muito (1+) skip

muito (1+) skip;

se mudarmos isto

(1 - x) = backslash

m 2 = true;

dificil se a dígitos

- Problema da escrita:

m 3 = false;

m 2 = false;

também escrita

- Problema da escrita:

mais de duas

muitas alternâncias entre o resultado e a modificação de cada vez que é feita!

é grande que a dígitos, volta

→ 1^o etapa: borrar o mover

	1	2	3
políptico de entretelas	while (true);	① while (true);	while (true);
	while (true) skip;	while (true) skip;	while (true) skip;

*

Toda dura desdoblado! Es solo un bucle. ① es, en el mismo tiempo del desdoblado, el bucle que hace; bucle 3 (A) el bucle

→ Una vez más queremos para el caso tanto una u m2 más guardando las otras last: paquetes que indican el último que entran

last = m1 = false; m2 = false;

while (last == false); while (true) no cumplen la condición de bucle =

que no cumple la condición de bucle =

while (true) {

 m1 = true;
 if (m1 == false) {
 last = true;
 return true;
 }

< m1 = (! m1 & ! last == false); >

 regla única;
 - el último que

 m1 = false;
 entra o salga

 regla única;

 while (true) {

 m2 = true;
 if (m2 == false) {
 last = true;
 return true;
 }

< m2 = (! m2 & ! last == false); >

 regla única;

 while (true) {

 regla única;

 if (true) con estos cambios solo se necesita una sola regla de cambio de cambio en cambio de cambio de cambio

 while (true) a veces = como bucle, bucle, bucle, bucle
 cambios como bucle, bucle, bucle

< m1 = (! m1 & ! last == false); >

 while (true) and last == false;

 while (true) and last == false;

< m2 = (! m2 & ! last == false); >

 while (true) and last == false;

 while (true) and last == false;

$\text{wh } m[1:n] = [n] \circ 0$, and $[1:n] = ([n] \circ 0)$;

space CS $i=2 \dots n$;

while (true) {

$m[i] = 0$;

 return m;}

$\text{for } j = 2 \dots n \text{ do } M[j] = \text{introduction value}$

$m[i] = j$; $M[j] = i$;

 print "for $i=1 \dots n$ such that $i \neq k$ " ;

 return m to print

 while ($m[k] \neq m[i]$ and $\text{last}[j] \neq i$)

$j = j + 1$

j can't be zero

 send i to j

$m[i] = 0$;

 return after message

 return m to return

m to print

$j = j - 1$;

$M[j] = \text{value}$

 return m to print

$j = j - 1$;

$M[j] = \text{value}$

 return m to print

$j = j - 1$;

$M[j] = \text{value}$

 return m to print

* Algorithm: $m[i] = \text{value}$

$m[i] = 0$;

return m to print

main(m: composition function)

$m[i] = \text{value}$ i such that $i \neq 0$ and $i \neq \text{length}(m)$

→ 1st iteration

polynomial of $m[1:n]$

while ($m[1:i] \neq 0$);

while ($m[1:i+1] \neq 0$);

polynomial of $m[1:n-1]$

$m[n-1] = 0$;

$m[n-1] = 0$ polynomial of $m[1:n-1]$

Non general function definition

$\{f\}(\text{value}) = \text{value}$

$\{f\}(\text{value}) = \text{value}$

$\{f\}(\text{value}) = \text{value}$

→ 2nd iteration: $\{f\}(\text{value})$

$\{f\}(\text{value}) = \text{value}$

value: composition function

$f(i) = \{f\}(\text{value})$ and $m[i]$

$m[1:i] = \text{value}$; $m[1:i+1] = \text{value}$;

$m[1:i+2:n] = \text{value}$ no message to print

$i = 1$

$i = 2$

$i = 3$

polynomial of $m[1:n]$

while ($m[1:i] \neq 0$);

while ($m[1:i+1] \neq 0$);

polynomial of $m[1:n-1]$

$m[n-1] = 0$;

$m[n-1] = 0$ polynomial of $m[1:n-1]$

spirals

$m[n-1] = 0$;

→ 3rd iteration:

predicates are entered

want1 = true;

while (want2) skip; // while (want2) skip

2 if (a <= b) want2 = true;

want2 = true; // want2 = true;

while (want2) skip; // while (want2) skip

+

predicates are entered:

skip = true;

3 if (a <= b) want2 = true; // want2 = true;

(a <= b) want2 = true; // want2 = true;

→ solution: don't use \neg because conflicts

will lead to

because the conflict

is a conflict

predicates are entered

want1 = true;

while (want2) {

want1 = false;

skip;

want2 = true;

want2 = true;

want1 = false;

want2 = false;

skip;

want2 = true; // want2 = true;

↓

atmosphere: if want2 does conflict → no take

conflict = 4

if (a <= b) want1 = true; //

want2 = true; // want2 = true;

+ want2 = false; //

want2 = true; // want2 = true;

if (a <= b) want1 = true;

want2 = true; // want2 = true;

predicates are entered: want1 = true;

want2 = true; // want2 = true;

want2 = false; //

want2 = true; // want2 = true;

if (a <= b) {

want2 = false; // want2 = true;

want1 = false;

want2 = true; // want2 = true;

while (a <= b) skip;

want2 = true; // want2 = true;

skip = true;

want1 = true; // want1 = true;

↓

if (a <= b) want1 = true;

want2 = true; // want2 = true;

want1 = false;

want2 = true; // want2 = true;

want1 = true;

want2 = true; // want2 = true;

spirals

spedire da sorgente turn = 0;
 -want 1 = false; want 2 = false;

* Ologramma: Mappa - Punto

want 1 = 0; want 2 = 0;

	1	2
spedire da sorgente	$\text{if } (\text{want 1} == \text{ok}) \text{ want 1} = 1;$ $\text{else want 1} = 2;$ $\text{while } (\text{want 1} == \text{want 2}) \text{ stop};$	$\text{if } (\text{want 1} == 1) \text{ want 1} = 0;$ $\text{else want 2} = 1;$ $\text{while } (\text{want 2} == \text{want 1}) \text{ stop};$
prendere da sorgente	want 1 = 0;	want 2 = 0;

se 1 spese entro want 1 = 0

B = 1 + 2 spese entro nel minor tempo?

al più vicino a 1/2 delle ore per il calcolo

da confrontare con 2 e 30 quindi confrontare con gli spese (o want 1 con 1° b' o 1)

una struttura dati per if è l'oggetto:

if 1

else 2

want 2 se 2 subito

else 1

want 1

una struttura dati do if è massima