

17.2.09.

For halting proof: [what is  $f_n$ ? & what is code?]

\* Java program - is a sequence of ASCII symbols.

\* In the computer, every ASCII symbol is a seq of 0's & 1's.

! \_ \_ ! \_ \_  $\leftarrow$  put 1 in front.

get a natural  $\#$ , this  $\#$  is called a code of the Java program.

what is  $f_c$ ?

let's start with a natural  $\# c$ .

we strip off 1 in front.

$f_c \rightarrow$  Java program corr. to  $\# c$ .

$$f(n) = \begin{cases} f_n(n) + 1 & \text{if } n \text{ is a valid code \& } f_n \text{ halts on } n. \\ 0 & \text{otherwise.} \end{cases}$$

[Not all computable function is  $\lambda$ -PR.

To prove this we will construct a function  $f(n)$ .

(a) which is computable.

(b) but not  $\lambda$ -PR.

(a) To prove the 1st part, our goal is to assign a  $n$ .

let's start with an expression like.  $PR(\pi^i, A(n))$ .

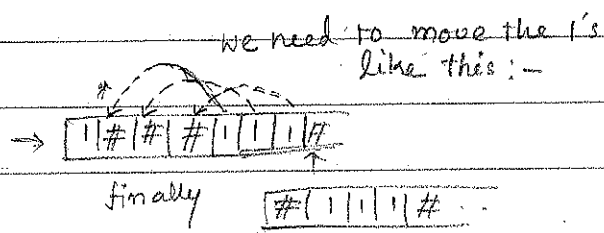
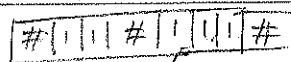
with Latex.

$$PR(\pi^{i-1}, A(n)).$$

Turning m/c :-

Projection:  $\pi_2^2 = (n_1, n_2) = n_2$ .

we want:



(start, #)  $\rightarrow$  (R, 1, erasing 1)

(erasing 1, 1)  $\rightarrow$  (erasing 1, #, R)

(erasing 1, #)  $\rightarrow$  (fwd 2, R)

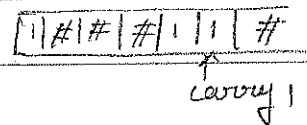
(fwd 2, 1)  $\rightarrow$  (fwd 2, R)

(fwd 2, #)  $\rightarrow$  (backtrack, L)

(backtrack, 1)  $\rightarrow$  (carry 1, #, L)

(carry 1, #)  $\rightarrow$  (check next blank, L)

(carry 1, 1)  $\rightarrow$  (carry 1, L)



(check next blank, 1) → (final cleaning, #, R)

(final cleaning, #) → (final move, 1, L)

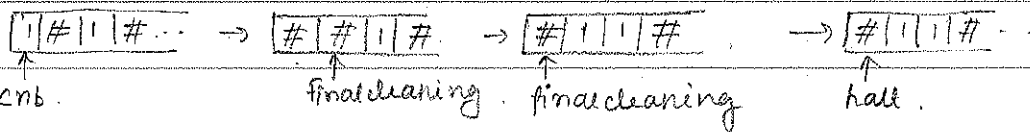
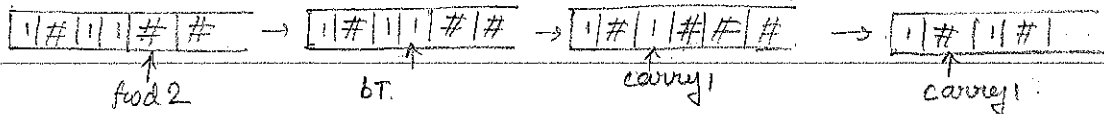
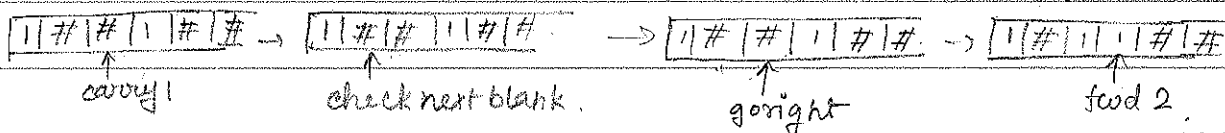
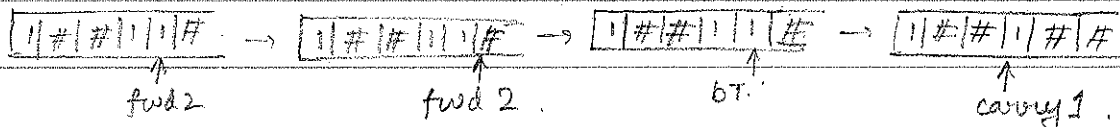
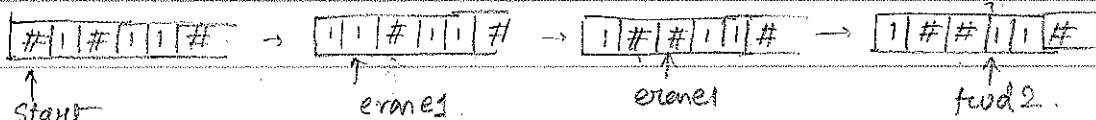
(final move, #) → halt

[order of these rules doesn't matter]

[hw:  $\pi^3_2$ ]

{ (check next blank, #) → (go Right, R)

{ (go Right, #) → (1, R, fwd 2)



cnb = check next blank