TD5: Producteur / Consommateur get of Cons Prod put MaiPloox Procl Cons On ne sait pas si les producteurs et les consommateur vont à la même vitesse QILL. struct mailbox } int data; sem-t plein; sem-t vicle; void mbox_init (struct mailbox * m) { sem-init (2m -> vide, 0, 1); sem-init (2m -> plein, 0, 0)

void mbox-pot/struct mailtox +m If int d) sem-wait 12m svide) m - data = d; som - past (& m = plein); int mbox-get/struct mailbox & m) sem-wait (2 m = plein). res = m -s data; sem - post (2 m -s vide). return res;

```
91.3 # define N
struct mailbox}
  int data[N];
   sem. t plein;
   sem-t vide;
   int head;
   int tail;
 void mbox-init (struct mailbox & m)
     sem-+12m -> plein, 0, N),
sem-+12m -> vide, 0, 0),
      m > head = 0;
m > tail = 0;
void mbox-put (struct sm, int d)?
      m -> clata Em -> head] = d;
      m = head = ((m -> head)+1) % N
       some post (2 m -> plein).
                        mailbox
int mbox-get (struct vs m) {
int rus;
      sem-wait 1 2 m -> plein).
       res = m = data[m=stail].
       m > tail = ((m > tail) + 1) /. N.
      return res;
```

91-4 struct mailbox 5 idem 93 sem - + mutex; tovjoursea un mutex void m box-init [idem Q1.3 sem-int (2 m = mutex, 0, 0). void mbox-put (______) som-wout (2 m -> viole) sem-wait (2 m -> mutex); m -> data Im -> head] = d; m -> head alm -> head)+1) / N; sem post (2 m -> mutex); sem-post (2 m -> plein) int res; sem - wait (2 m -> plain); sem - wait (2 m -> mutex) sem - wait (2m -> mutex). sem-post (2 m-s viole); return res