Public Exclusion:	Loke 1 0
Lock => 0	
	2 (our) slide
while (true) {	) gove has nevy wolide
	- and-swap (& lock, 1, 1) 1, 20);
	Market and the second s
1 * d	to nothing *1 * printer do
92 17 CS *	
	I* CVIHCON Section*/
	10de=0;
3	
E THE STREET	/* Remainder Section*1
	· Lock > 0
	£
po(a), polaz) Cslanos	8 (8)   0! = 1   7 F
pi (a1), pi (a2) CS	Not satisfied, it runs 2 process in
Alle (Maril Comment) of A Mil	the Cs at the same time.
2 progress: Loc	IX > 0,000 own of then
	Look will move change, and
po(a1), po(a2) CS, po(	a3), po (a4) RS satisfied
p, (a,), p, (a2) CS, p, (	a3), p1 (a4) RS
P1 (a1), po (a2) cs.	
	- December and of
3) Bounded waiting:	Lock -> BAO
po (a) waiting	Not satisfied, Because he second process
	entered it's cs, before po.
p1 (a1), p1 (a2) CS	

Lock => 1	1 Mutual Exclusion:
the second second second second second second second	Lock => 1 0 = 4 => 1
while (true) {	
as white (compare_and=swap (	po(a1), waiting and of the
(8 lock, 0, 1) (20);	
To see upon second Francisco the	
1 do nothing */	why? 1 = 0 => return   1
La summer hours and extent to	11 = 0
92 1* Critical Section*/	=> (T) obod
93 lock =0;	2 progress:
the second second selection of the second	The same of the sa
94 /* Remainder Section */	po(a1), waiting
	1 Mutual Exclusion:
3	
	3 Bounded waiting:
- This algorithm will loop	HOM STORY MAN
infinitely in the (a) state.	spo(a1), waiting
nent be he value of the	E progress:
Lock will never change, and	
	a decided to the second
with it being 1, no process	+
with it being 1, no process	
can enter.	
can enter.	
- Doesn't satisfy any of	3) Bounded waiting: 1
can enter.	es : priling Labour (8
- Doesn't satisfy any of the three requirements.	
- Doesn't satisfy any of the three requirements.	- 416
- Doesn't satisfy any of the three requirements.	

Lock => 0	1 Mutual Exclusion:
<u> </u>	Lock → Ø, 1, 1 F, F, F, F
a, waiting [i] = 1;	key - 1,0,1 FT
92 Key = 1;	
as while (waiting [i] && key)	po(a1), po(a2), po(a3), po(a4)
94 key = test_and_set (&lock);	TOT
as waiting [i] = 0;	p1(a1), p1(a2), p1(a3), p1(a4)
as 1* Critical Section*/	waiting.
a7 j= (i+1) /. n;	
as while ((j! zi) && (!waiting [j]))	2 progress:
ag j= (j+1)%u;	Lock > Ø,1 R, F, F, F
quoif (j = =i)	key → X, O F
an lock 20;	
else	po(a1), po(a2), po(a3), po(a4), po(a3)
a12 waiting [j] 20;	pu(as), pu(a6) (S, pu(a7) j= 0+1%4=
a13 /* Remainder Section*/	po(as), po(a6) (S, po(a7) j=0+1%4=1 T&T T&T po(a8), po(a9) j=1+1 /64=2, po(a8),
3 while (true)	po(aq) j= 2+1 1/4=3, po(aq), po(aq)
The state of the s	
- This algorithm satisfies	- The loop continues without the
Mutual Exclusion, and boom	
ed waiting, But doesn't	3 Bounded weiling:
satisfy progress.	Lock → Ø, X, 1 F, F, F
	key → X, Ø, X, 1 TT
why not?	
~~~	po(a1), po(a2), pu(a3), po(a4),
Because He loop will loop	T&F po (as) po(as) -> waiting.
infinitely without the statemer	T P T
as becoming (F).	P1(a4), P1(a3), P1(a4)
	waiting