dequeue 2. A. Procedure Willy William. (in/aut Q: Queue, out P:odr) komis function is Empty (Queue) -> bookan Algoritma P = Q. hevd if (not :s Empty(Q)) then if Q. head = Q. to:l then Q. head = P. > rext Q. head = Prev = NIL P > next = ML Q. to:l = NIL endif endif endif endif endif endif function :s Empty (Queue) -> bookan Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (not is Empty (Q. Reg.)) then if (not is Empty (Q. Reg.)) then for i = 1 to 3 do dequeue (Q. Reg., P) if P!= NIL then enqueue (Q. Covid, P)	Made Narodeon Handika Pramesta/103032300101	
Kownus Function is tempty (Queue) -> boolean Algoritma P = Q. head if (not is Empty(Q)) then if Q. head = Q. too! then Q. head = P-> rext Q. head => prev = NIL P-> next = ML Q. too! = NIL Q. too! = NIL Q. too! = NIL Q. too! = NIL Procedure pindah Tast Covid (in/aut Q. Reg., Q. Covid : Queue) Kamus P: Odr i: integer function: is Empty (Queue) -> boolean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is tripty (Q. covid)) then if (not is Empty (Q. covid)) then	Lamena	
Kownus Function is tempty (Queue) -> boolean Algoritma P = Q. head if (not is Empty(Q)) then if Q. head = Q. too! then Q. head = P-> rext Q. head => prev = NIL P-> next = ML Q. too! = NIL Q. too! = NIL Q. too! = NIL Q. too! = NIL Procedure pindah Tast Covid (in/aut Q. Reg., Q. Covid : Queue) Kamus P: Odr i: integer function: is Empty (Queue) -> boolean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is tripty (Q. covid)) then if (not is Empty (Q. covid)) then	2. A Procedure MMMMenn. ("Yout Q: Queve, out P:odr)	
Function is Empty (Queue) -> boolean Algoritma P = Q. head if (not is Empty(Q)) then if Q. head = Qtoil then Q. head = P-> rext Q. head => prev = NIL P-> next = ML else Q. head = NIL Q. toil = NIL Q. toil = NI endif endif endprogram 2C. Procedure pindah Tast Covid (in/out Q.Reg., Q.Cavid: Queue) kamus P: odr i: integer function: is Empty (Queue) -> boolean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Q.covid)) then if (is Empty (Q.covid)) then		
Algoritma P = Q.heud if (not is Empty(Q)) then if Q.head!= Qtail then Q.head = P+rext Q.heud => prev = NIL P>rext = ML else Q.heud = NIL Q.tail = NIL endif i integer function is Empty (Queue) > bodean frocedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcould)) then if (is Empty (Qcould)) then	Function is Egypty (Queue) -> boolean	
P = Q.hevd if (not is Empty(Q)) then if Q.head != Q.tail then Q.head = P-> rext Q.hevd -> prev = NIL P-> next = ML Q.toil = ML Q.toil = ML Quit Qui		
P = Q.hevd if (not is Empty(Q)) then if Q.head != Q.tail then Q.head = P-> rext Q.hevd -> prev = NIL P-> next = ML Q.toil = ML Q.toil = ML Quit Qui	Algoritma	
if Q.head != Q.tail then Q.head = P>rext Q.head = P>rext Q.head = Prext Q.head = NIL P>rext = ML Q.head = NIL Q.tail = ML Q.tail = Q.tail Q.tail = ML Q.tail = ML Q.tail = ML Q.tail = ML Q.tail = Q.tail Q.tail =	P = Q. head	
if Q.head != Q.tail then Q.head = P>rext Q.head = P>rext Q.head = Prext Q.head = NIL P>rext = ML Q.head = NIL Q.tail = ML Q.tail = Q.tail Q.tail = ML Q.tail = ML Q.tail = ML Q.tail = ML Q.tail = Q.tail Q.tail =	if (not is Empty(Q)) then	
Q.head = P=rext Q.head = Prev = NIL P=rext = ML Q.head = NIL Q.toil = NIL Q.toil = NIL Endif Endprogram 2C. Procedure pindah Test Cavid (in/aut Q.Reg., Q.Cavid: Queue) Kamus P: Odr i: integer function: Empty (Queue) -> badean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma If (is Empty (Q.cavid)) then id (not): Empty (Q.Real) Lhen	if Q. head != Q. tail then	
P= next = ML else Q. head = NIL Q. toil = NL endif endif endif endif endif endif conductor gindah Tast Covid (in/out Q.Reg., Q.Cavid: Queue) kamus P: Odr i: integer function is Empty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Q.covid)) then if (not is Empty (Q.covid)) then	Q-head = P-> next	
endif endif endif endif endrogrown 2. Procedure pindoh Test Covid (in/aut a Reg. a Covid: aveue) kamus P: odr i: integer function: s Empty (aveue) -> badeon Procedure enqueue (aveue, adr), Procedure dequeue (aveue, adr) Algoritma if (is Empty (acavid)) then if (not is Empty (acavid)) then	Q.heud -> prev = NIL	
Q. head = NIL Q. toil = NI Endif Endif Endprogram 2.C. Procedure pindah Tast Covid (in/aut Q.Reg., Q.Cavid: Queue) Kamus P: Odr i: integer function: s. Empty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Q.cavid)) then if (vat is Empty (Q.Reg.) then	P-> next = ML	
Q.to: = Ml endif endif endif endprogram 2.C. Procedure pindah Tast Covid (in/aut Q.Reg., Q.Cavid: Queue) kamus P: odr i:integer function: s Empty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Q.cavid)) then if (not is Empty (Q.Rea)) then	else	
endif Endprogram 2.C. Procedure pindah Test Covid (i Maut QReg, QCavid: Queue) Kamus P: Odr i: integer function: SEmpty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcavid)) then if (not is Empty (QRea)) then		
endprogram 2.C. Procedure pindah Test Covid (in/out QRieg, QCovid: Queue) Kamus P: odr i: integer function: SEmpty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcovid)) then if (not is Empty (QRea)) then		
endprogram 2C. Procedure pindah Test Covid (in/out a Rieg, a Covid: aveve) Kamus P: odr i: integer function: SEmpty (aveve) -> bodean Procedure enqueve (aveve, adr), Procedure dequeve (aveve, adr) Algoritma if (is Empty (acovid)) then if (not: Empty (acovid)) then		
2C. Procedure pindah Tast Covid (in/out a Rieg, a Cavid: aveve) Kamus P: odr i: integer function: s Empty (aveve) -> bodeon Procedure enqueve (aveve, adr), Procedure dequeve (aveve, adr) Algoritma if (is Empty (acavid)) then if (not: s Empty (acavid)) then		
Kamus P: Odr i: integer function: is Empty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcovid)) then if (not is Empty (QRea)) then	enaprogram	
P: Odr i: integer function: is Empty (Queue) -> bodeon Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcovid)) then if (not is Empty (QRea)) Lben	2C. Procedure pindah Tast Covid (in/out a Rieg, a Cavid: Queue)	
i : integer function :s Empty (Queue) -> bodean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcovid)) then if (not is Empty (QRea)) Lbon		
function is Empty (Queue) -> boolean Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma if (is Empty (Qcovid)) then if (not is Empty (QRea)) then		
Procedure enqueue (Queue, adr), Procedure dequeue (Queue, adr) Algoritma If (is Empty (Qcovid)) then If (not is Empty (QRea)) then		
Algoritma if (is Empty (Qcovid)) then if (not is Empty (QRea)) then	Punction is empty (autre) - state on	
if (is Empty (Qcovid)) then if (not is Empty (QRea)) Lben		
if (not is Empty (a Reg)) then for i=1 to 3 do	in (is Empty (Qcould)) then	
for i=1 to 3 do	if (not interpret 10 Rea) Llon	
0 121 00 300	For i - 1 to 2 le	
do que (ORos P)	de que (ORea P)	
in Pl= NIL than	in Pl= NIL +bon	
Produce (QCoud P)	Prayoue (QCoud P)	
end if	entil	
littor	lmfor	
endif	lndif	
else '	else '	
output ("pemindahan belum bisa di latutan")	Output ("Pemindahan belum bisa dilatutan")	
2: 10:0	2: 10.0	
end procedure	endprocedure	
(STOIL)	(STOIL)	

b. `	Procedure enqueue ("Yout Q: Queue, in P:	odr)
	CUMUS	
	function is Empty (Queue)	
	Hapritma (S)	
	if is Empty (Q) then Q head = P Q toil = P	
	R head = P	
	Q toil = P	
	e (go	
	P-> prev = Q.tail	
	Q tail = next = P	
	$\Omega + rad = P$	
	end f	
. 6	Mprocedure	
,		
		• •
-		