Practica 1

1. a)

int M 
int N 
5; 
int numeros[M] ; 
int numero 
int total 
N; 
process buscador [i 
if (numeros[i] 
<total++; > 
to M]{ 

3)



Process Productor: . 
{ while (true) 
{ produce elemento 
<await (cant < N)> 
buffer [pri_vacia] = elemento; 
pri_vacla 
(pri_vacia + 1) mod N; 
Process Consumidor: . 
{ while (true) 
{ <await (cant > 
elemento = buffer [pri_ocupada] ; 
<cant--> 
(pri_vacia + 1) mod N; 
pri_vacla 
consume elemento 



int 
int 
int 
int 
int 
int 
P 
5; 
cant = e; 
pri_ocupada = 0; 
pri_vacia = e; 
buffer 
Process Productor[i = to P] 
{ while (true) 
{ produce elemento 
<await (cant < N); 
buffer [pri_vacia] = elemento; 
pri_vacla 
(pri_vacia + 1) mod N; > 
Process Consumidor[i = to C] 
{ while (true) 
{ <await (cant > 0); 
elemento = buffer [pri_ocupada] ; 
<cant--> 
(pri_vacia + 1) mod N; > 
pri_vacla 
consume elemento 



c) 
int personas 
boolean enUso = false; 
personas cola; 
boolean ready [n] = false; 
int siguiente 
process personas [i 
1 to n]{ 
ready [i ] 
true; 
<await i 
siguiente; > 
imprimir(documento) ; 
ready[i] = false; 
process coordinador{ 
while(true){ 
int j 
1; 
while( && j<=n){ 
if(j < n){ 
siguiente = j; 
<await ready[siguiente] 
false> 



int pila[5]; 
Process usador[i = to 5] 
{ while (true) 
{ <await (pila. length > e); 
elem = pila.pop(pila);> 
usa el elemento; 
<pila . push (elem) ; > 

 5)

a) 
int personas 
boolean enUso 
false; 
process personas [i 
1 to n] { 
<await ! enUso; enUso = true> 
Imprimir(documento) ; 
unUso = false; 
b) 
int personas 
boolean enUso 
personas cola; 
int siguiente 
false; 
int proximo = 1; 
process personas [i 
1 to n] { 
<int id 
siguiente + 1; 
<await id 
proxlmo> 
imprimir(documento) ; 
P OX I mo-H_ ; 
c) 
int personas 
boolean enUso 
personas cola; 
false; 
boolean ready [n] = false; 
int siguiente 
process personas [i 
1 to n]{ 
ready [i] = true; 
<await i 
siguiente; > 
imprimir(documento) ; 
ready[i] = false; 
process coordinador{ 
while(true){ 
int j 
1; 
while( && j<=n){ 
if(j < n){ 
siguiente = j; 
<await ready[siguiente] 
false> 
d) 
int personas 
boolean enUso 
personas cola; 
int siguiente 
false; 
int proximo = 0; 
process personas [i 
1 to n] { 
<int id 
siguiente + 1; 
<await id 
proxlmo; > 
imprimir(documento) ; 
enUso = false; 
process coordinador{ 
while(proximo < n) 
enUso = true; 
p r oxlmo-++ ; 
<await ! enUso> a) 
int personas 
boolean enUso 
false; 
process personas [i 
1 to n] { 
<await ! enUso; enUso = true> 
Imprimir(documento) ; 
unUso = false; 
b) 
int personas 
boolean enUso 
personas cola; 
int siguiente 
false; 
int proximo = 1; 
process personas [i 
1 to n] { 
<int id 
siguiente + 1; 
<await id 
proxlmo> 
imprimir(documento) ; 
P OX I mo-H_ ; 
c) 
int personas 
boolean enUso 
personas cola; 
false; 
boolean ready [n] = false; 
int siguiente 
process personas [i 
1 to n]{ 
ready [i] = true; 
<await i 
siguiente; > 
imprimir(documento) ; 
ready[i] = false; 
process coordinador{ 
while(true){ 
int j 
1; 
while( && j<=n){ 
if(j < n){ 
siguiente = j; 
<await ready[siguiente] 
false> 
d) 
int personas 
boolean enUso 
personas cola; 
int siguiente 
false; 
int proximo = 0; 
process personas [i 
1 to n] { 
<int id 
siguiente + 1; 
<await id 
proxlmo; > 
imprimir(documento) ; 
enUso = false; 
process coordinador{ 
while(proximo < n) 
enUso = true; 
p r oxlmo-++ ; 
<await ! enUso> 

* 1. La condición que no cumple es la ausencia de demora innecesaria, ya que si el primer proceso tiene que entrar 2 veces seguidas no va a poder porque tiene que esperar al proceso dos.

Texto alternativo generado por el equipo:
int arribo[l:n] 
int continuar[l : 
processWorker[i = 1 to n]{ 
while(true){ 
arribo[i]= 1; 
while (continuar[i] 
a) skip; 
/ [hace mambos críticos 
continuar[i] = e; 
processCoordinador { 
while(true){ 
for[i 
1 to n st arribo[i] 
arribo[i]= e; 
continuar [i] 
1; 
while(continuar[i] 
e) skip; 