```
#include<stdio.h>
#include<unistd.h>
#include<semaphore.h>
#include<stdlib.h>
int guard = 0;
int semaphore value = 0;
wait ()
 {
while (TestAndSet(&guard) == 1);
 if (semaphore value == 0)
 {
 atomically add process to a queue of processes waiting for the semaphore and set guard to 0;
}
 else
{
semaphore value--; guard = 0;
}
 }
signal()
{
while (TestAndSet(&guard) == 1);
if (semaphore value == 0 && there is a process on the wait queue) wake up the first process in
the queue of waiting processes
else
semaphore value++;
 guard = 0;
}
```

Ans 2

```
#include<stdio.h>
#include<unistd.h>
type resource=monitor var P:

array[0..2]of boolean;

X:condition;

procedure acquire(id: integer, printer-id: integer);

begin

if P[0]and P[1]and P[2]then X.wait(id)if

 not P[0]then printer-id:= 0;

else if not P[1]then printer-id:= 1;

else printer-id:= 2;
```

```
P[printer-id] :=true;end
procedure release(printer-id: integer)
begin
P[printer-id] :=false;X.signal;end
Begin
P[0] :=P[1] :=P[2] :=false;end
```