Assignment -5

OPERATING SYSTEM

Solutions

1. Which of the below algorithms which implements synchronization mechanisms and satisfies all the four conditions such as Mutual Exclusion, Progress, Bounded Waiting and User Mode Execute?

(a) Random selection
(b) TSL Algorithm
(c) Peterson's
(d) All of the above

Solution: Option (c)

2. var occupied
var blocked
Enter Region:
{
If (occupied) { then
blocked= blocked +1
sleep ( );
}
else occupied= 1;
}
Exit Region:
{ occupied= 0 If
(blocked) { then
wakeup (process);
blocked= blocked – 1;
}
}

(1) Mutual Exclusion is guaranteed
(2) Deadlock free Algorithm
(3) Progress is guaranteed

Which of the above statements are False?
(a) Only 1
(b) 1 & 2
(c) 1 & 3
(d) none of the above

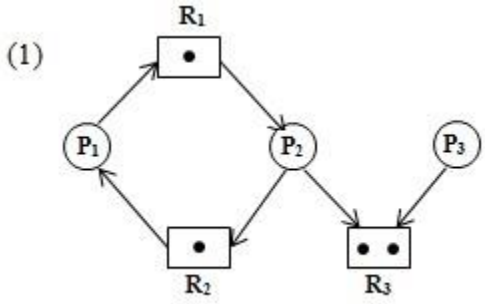
Solution: (d)
Explanation:
As the initial values are not declared for both the variables

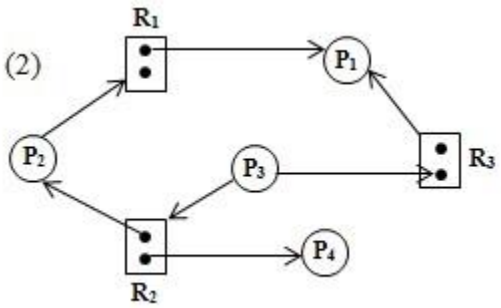
3. If ‘m’ processes share ‘n’ resources of the same type, the maximum need of each process does not exceed ‘n’ and the sum of all their maximum needs is always less than ‘m+n’. In this case:

(a) Deadlock can never occur
(b) Deadlock may occur
(c) Deadlock has to occur
(d) None

Solution: Option (a)

4. Which of the following is True?





(a) Deadlock occurs in both the cases

(b) Deadlock occurs in (1) but not in (2)

(c) Deadlock occurs in (2) but not in (1)

(d) None of the above

Solution: Option (d)

5. Consider the below two threads:

thread A: lock 1, lock 2, unlock 2, unlock 1

thread B: lock 2, lock 1, unlock 1, unlock 2

(a) Deadlock may occur
(b) Deadlock has to occur
(c) Deadlock will not occur
(d) None

Solution: Option (a)

6. Consider the following program:

Semaphore $M_x = 1$, $M_y = 0$

Void P( ) Void Q( )

```
{
  while (1)
  {
    P(Mx);
    Print '0';
    V(My);
  }
}
```

```
{
  while (1)
  {
    P(My); P(Mx);
    Print '1';
    V(Mx);
  }
}
```

The possible output of the above program:
(a) Any number of zeros followed by any number of 1's
(b) Any number of ones followed by any number of 0's
(c) Zero followed by deadlock
(d) One followed by deadlock

Solution: Option (c)

7. P_0: P(S), P(Q), Print("Hello"), V(Q), V(S)
P_1: P(Q), P(S), Print("Hi"), V(Q), V(S)
Where S & Q are two semaphores initialized to 1.

In the above situation:
(a) Deadlock may occur
(b) Bounded Waiting is satisfied
(c) Deadlock never occurs
(d) Both (a)&(b)

Solution: Option (a)

8.
```
#define N 4
void philosopher(inti)
   {
while(true)
        {
think( );
take_fork(i);
take_fork((i+1) %
N); eat( );
put_fork(i);
put_fork((i+1) %
N);
          }
}
```

(a) At any point of time atleast one philosopher will be eating
(b) Deadlock has to occur
(c) Deadlock may occur (d) Both (a) & (c)

Solution: Option (c)

9. Which of the classical IPC problem is not solved using semaphores?

(a) Producer Consumer
(b) Dining Philosopher
(c) Sleeping Barber
(d) None of the above

Solution: Option (d)

10. Consider a concurrent program with two processes P & Q, where A, B, C, D & E are arbitrary atomic statements; assume that main program does a Par begin of the two processes:

```
Void P( )
{
  A;
  B;
  C; }
        }
```

```
Void Q( )
{
D;
E;
```

Which of the following statement is True?
(a) It is never possible to generate output 'C' before output 'E'
(b) The order of Execution of statements will be same independent of the process execution (c) The relative order among the statements of P & Q is always maintained (d) None

Solution: Option (c)

11. Consider the following program:

```
Constint n=
10 int Count=
0
Void
A( ) {
int i;
for(i= 1
to n)
  Count= Count + 1;
}
```

```
Main ( )
{
   Par begin
A( );
A( );
A( );
A( );
   Par end
}
```

What is the minimum and maximum possible value of count after the completion of the program?

(a) 1, 40 (b) 2, 40
(c) 3, 40 (d) 4, 40

Solution: Option (b)

12. A counting Semaphore is initialized to 20 and then 12 down operations and 8 up operations were completed on the Semaphore. The resulting value is:

(a) 16 (b) 12
(c) 20 (d) 14

Solution: Option (a)

13. The Semaphore variable full, empty and mutex are initialized to 0, n and 1 respectively. Process P_1 repeatedly adds one item at a time to a buffer of size n, and process P_2 repeatedly removes one item at a time from the same buffer using the programs given below. In the programs, K, L, M & N are unspecified statements.

```
P1: while(1) {
      K;
      P(Mutex);
      Add item to buffer;
      V(mutex);
      L;
      }
P2: while(1)
     {
     M;
     P(mutex);
     Remove item from the buffer;
     V(mutex);
```

N;
}

These statements K, L, M & N are respectively:
(a) P(full), V(empty), P(full), V(empty)
(b) P(full), V(empty), P(empty), V(full) (c) P(empty), V(full), P(empty), V(full)
(d) P(empty), V(full), P(full), V(empty)

Solution: Option (c)

14. Consider two processes P_1 and P_2 accessing the shared variables x and y protected by two binary semaphores S_x and S_y respectively, both initialized to 1. P and V denote the usual semaphore operators, where P decrements the semaphore value, and V increments the semaphore value. The pseudo-code of P_1 and P_2 is as follows:

P_1:	P_2:
While true do {	While true do {
L_1: ……	L_3: ……
L_2: ……	L_4: ……
X= X+1;	Y= Y+1;
Y= Y-1;	X= X-1;
V(S_x);	V(S_y);
V(S_y);	V(S_x);

In order to avoid deadlock, the correct operators at L_1, L_2, L_3 and L_4 are respectively:
(a) P(S_y), P(S_x); P(S_x), P(S_y)
(b) P(S_x), P(S_y); P(S_y), P(S_x)
(c) P(S_x), P(S_x); P(S_y), P(S_y)
(d) P(S_x), P(S_y); P(S_x), P(S_y)

Solution: Option (d)
15. The time taken to switch between user and kernel modes of execution be t_1 while the time taken to switch between two processes be t_2.

Which of the following is TRUE?
(a) $t_1 > t_2$
(b) $t_1 = t_2$
(c) $t_1 < t_2$
(d) nothing can be said about the relation between t_1 and t_2

Solution: Option (c)

Explanation:
Process switching involves mode switch. Context switching can occur only in kernel mode.

16. A solution to the Dining Philosopher's problem which avoids Deadlock can be:

(a) Ensure that all the Philosopher's pick up the left fork before the right fork
(b) Philosophers can select any fork randomly
(c) Ensure that all the Philosophers except one pick up the left fork while that particular philosopher pick up right fork before left fork (d) Deadlock cannot be avoided

Solution: Option (c)

17. Consider the below program as two concurrent processes:

Semaphore x:= 0;

P_1: repeat forever	P_2: repeat forever
V(x);	P(x);
Compute;	Compute;
P(x);	V(x);

Consider the following statements about process P_1& P_2:
(1) It is possible for process P_1 to starve.
(2) It is possible for process P_2 to starve.

Which of the following is true?
(a) Both 1 & 2 are true
(b) Both 1 & 2 are false
(c) 1 is true & 2 is false
(d) 2 is true & 1 is false

Solution: Option (a)
18. In order to allow only one process to enter the Critical Section, binary semaphore is initialized to:

(a) 0
(b) 1
(c) 2
(d) 3

Solution: Option (b)

19. Minimum no. of processes required for deadlock is:

(a) 4
(b) 10
(c) 2
(d) Infinite

Solution: Option (c)

20. Semaphore is used to enforce Mutual Exclusion and Synchronization between processes interacting over shared data and variables. Which of the following statements is true about semaphores in this regard?

(a) The Operations SIGNAL(S) & WAIT(S) needs to be atomic.
(b) A process exiting the CS will call SIGNAL(S)
(c) 'Busy-Wait' solutions to the Critical Section are typically implemented using machine instructions that execute in the Kernal mode

(d) all of the above

Solution: Option (d)