Task 1

This program might deadlock if both processes hold one lock and need the other at point np(lock2) , P(lock1). each of these processes executes their first P task and is directed to the second. Now both of these will wait forever in deadlock

Deadlock state final values:
x == 2, y == 1, z == 2

if terminated:
protected by lock1 x will be 3
protected by lock2 y == 3
but z == 1,2 or 3 and is not protected by any lock. Usual error if the program terminates is z will only have the value 3.

Task 2:

in the function go
we can see that the then statement is the data that is gathered from the promise function ShowCircle. In a promise a then case is where the input given to the function is the return value of a promise
here:
div is a parameter that holds the return value of ShowCircle

We need to return a new promise, or we'll get undefined in the go function promise because it will be undefined and does not have a result value or state. The result state is initially undefined, so we need a promise to always return a value or an error if the promise was rejected.

The result starts it's state as pending and result as undefined. Then when eventually resolve is called the state will be fulfilled and the result will be the value.

setTimeout is used to let us know how long the process should last before the function calls resolve, which as mentioned changes the state of the promise, and produces the result.

The meaning of the lines 50-53 indicates that the state of the promise and result production is initialized by an event.

The 0 is an indicator of the start in a transitioning event. This means that the event would transition to 1, and this is the end phase.

Task 3:

Mark which of these are true and which are false. Each correct answer gives 1 point. Points are not deducted for wrong answers.

1. "In monitors, condition synchronization is provided implicitly."
    a. False

2. Consider the following code block: int x = 0; int y = 0; co x = y + 1; || y = x + 1; oc. Does the first assignment (x = y + 1) satisfy the At-Most-Once property?

    True

3. "In Signal-and-Continue signaling discipline for monitors, the process executing signal passes control of the monitor lock to the process awakened by the signal."
    a. True

4. "In the Await Language, operation signal(cv) applied to condition variable cv has no effect if the delay queue of cv is empty."
    a. True

5. "In Signal-and-Wait signaling discipline for monitors, the signaler continues and the signaled process executes at some later time."
    a. False

6. "In asynchronous message passing, receive is a blocking primitive."
    a. True

7. "A state change caused by an atomic action is indivisible and hence cannot be affected by atomic actions executed at about the same time."
    a. True

8. "When emulating monitor-based programs with message-based programs, wait statement corresponds to saving a pending request."
    a. True

9. "A parallel execution can be modelled as a linear history, because the effect of executing a set of atomic actions in parallel is equivalent to executing them in some serial order."
    a. True

10. "In the Await Language, operation empty(cv) applied to condition variable cv clears the queue of cv."
    a. True

11. "Mutual exclusion is an example of a liveness property."
    a. False

12. "Termination is an example of a safety property."
    a. False

13. "A liveness property is one in which the program always enters a good state, i.e., a state in which variables have desirable values."
    a. True

14. "A fine-grained atomic action is one that is implemented directly by the hardware on which a concurrent program executes."
    a. True

15. "Java uses Signal-and-Continue signaling discipline for monitors."
    a. True

16. "Eventual entry to a critical section is an example of a safety property."
    a. False

17. "In the AWAIT language, the await statement can only be used to specify fine-grained atomic actions."
    a. False

18. "Partial correctness is an example of a liveness property."
    a. True

19. "The state of a concurrent program consists of the values of the program variables at a point in time."
    a. True

20. "A scheduling policy is unconditionally fair if every unconditional atomic action that is eligible is executed eventually."
    a. True