Operating Systems CO-562

Abumansur Sabyrrakhim
Assignment 3

Problem 3.1

(a)

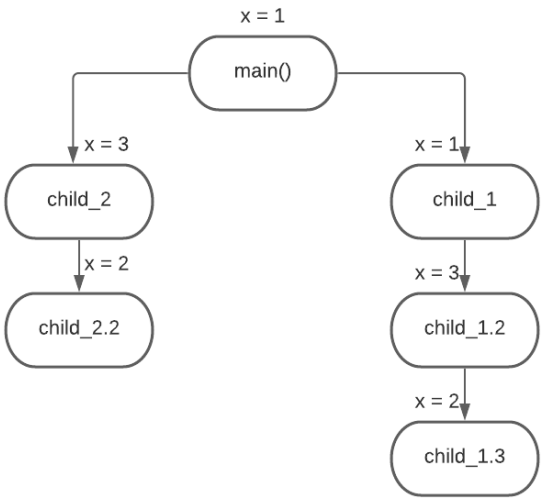

Figure 1: The diagram above shows all the processes during the program's execution

(b)

the output:

p0: x = 1
p1: x = 1
p2: x = 3

p3: x = 3
p4: x = 2
p5: x = 2

Problem 3.2

(a)

In case we make first call to reader() process, lets consider that reader number 1 (r1) wants to read and calls reader() process.

First it executes down(mutex), which means it changes the initial value of mutex from 1 to 0, so now mutex = 0;

Then, it updates readcount by 1, so now readcount = 1;

Next we have down(writer) if readcount = 1, which means if reader is present, writer cannot enter the critical section. So, now writer = 0;

Then we perform up(mutex), such that now mutex = 1;

Now while r1 starts reading data, reader number 2 (r2) wants to enter the critical section, it executes, and since both of the readers are in critical section so now mutex = 1 and readcount = 2. Now, writer number 1 (w1) wants to enter critical section and wants to perform write() process, but since readers r1 and r2 are present, w1 is not allowed to enter. Now r1 wants to leave, so down(mutex) is perfomed, so readcount is 1 and mutex is 0 now . After that we perform up(mutex), and update value of mutex to 1. Writer 1, w1, still cannot enter the critical section, since the value of readcount is 1. Now, r2 wants to leave critical section, so mutex = 0, readcount = 0. Afterwards, we perform up(mutex), and now mutex is equal to 1. The issue here is that any other reader can enter the critical section, but no writer can do so.

(b)

In this case, everything is similar to case (a) before r2 wants to leave. So, when r2 wants to leave, it goes to if(readcount == 0), but the value of readcount is still 1, so it goes to else part, and updates the value 0f mutex by one (up(mutex)), and then r2 is leaving, so now down(mutext) and readcount–, so now both, mutex and readcount are equal to 0. And now, since readcount = 0, it goes to if condition, and performs up(mutex) and before executing up(writer), w1 calls writer() process again, but cannot enter the critical section, that is the loophole of this case's reader() function.

(c)

Lets consider that r1 wants to enter critical section, r1 calls function read(), updates mutex to 0, readcount to 1, and then goes to if condition, since readcount = 1, we set value of writer to 0 (down(writer)), and value of mutex to 1 (up(mutex)). Now, r1 wants to leave and performs exit by setting mutex and readcount to 0. Then, it executes if (readcount == 0) condition, and performs up(writer), so now writer = 1. At the same time, w1 calls writer() function, and performs down(writer), s.t. now it's 0, and down(mutex), but since mutex = 0, w1 cannot enter, which is loophole of the writer() function.

Problem 3.3

Please see code attached.