**CS2106 Operating Systems**

**Assignment 2 – Processes and Threads**

**Answer Book**

|  |  |
| --- | --- |
| Member 1 Name: Ram Janarthan | Member 1 Matric No: A0147620L |
| Member 2 Name: Advay Pal | Member 2 Matric No: A0144939R |

Question 1

This is what I see on the screen: Parent sent message: Hello child! and 128

My single statement description is: This program allows a parent process to send a message to child process via a unix pipe.

Question 2

The sizeof function returns the size (in bytes) of the argument passed to it.

Question 3

My completed code is attached below:

//ATTACH CODE

Question 4

The threads print out of order. The reason is that threads can get pre empted before the execute the print statement.

Question 5

The threads do share memory. Referring to ctr, I conclude this because the value of ctr in different cases is not just 0/1, but also 2,3...8,9.

Question 6

The values of ctr as printed by the threads are wrong. The reason is sometimes the threads are pre-empted before they can increment ctr.

Question 7

The variable "i" must be cast into void \* because that is the argument type for the start routine of pthread\_create. This is to ensure that the argument can be of any type, as void\* represents the type of a generic pointer. So we can simply pass in the address no matter what type the argument is.

In the child it does not have to be cast back into int because the %d qualifier tells the compiler to implicitly typecast the pointer as an integer.

Question 8

The changes I made are to include a ‘pthread\_join’ call to the ‘i’th thread immediately after it has been created. This ensures that no thread ‘i’ gets created before the ‘i-1’th thread has been executed.

My code is attached here: (On line 25)

pthread\_join(thread[i], NULL);

Question 9

The value of glob printed at the end of main is 20

Question 10

The changes we made are as follows:  
Change child((void \*) i); to   
pthread\_create(&thread[i], NULL, child, (void \*) i);  
  
and add the following line to the end of the child function  
pthread\_exit(NULL);

Question 11

The values of glob are incorrect because all threads share the same glob variable and threads are preempted at random points during the execution, so we cannot guarantee that one thread manages to increment glob before it is pre empted. Furthermore, we see that no process manages to reach it’s exit print statement, which we attribute to the fact that it sleeps for 1s, which means it is still asleep by the time the main function is done.

Question 12

The threads still update glob incorrectly, with the only difference that glob is either 0 or 1. This is because a thread may get pre empted before it locks the mutex, or after it locks the mutex and increments glob. If it gets pre empted after locking the mutex, but not before unlocking, no other thread can increment glob. Again, we don’t see any thread unlocking the mutex, which we attribute to 1s being too long a duration.

Question 13

The changes we made were as follows:  
1. Delete all lines dealing with the mutex  
2. Add a pthread\_join statement immediately after the pthread\_create statement

Our program is attached below:

//ATTACH CODE