**DESCRIPTION**

Q41. Input: N - Number of primes, K - Number of child processes. Take these as arguments. Output: List of prime numbers Multiple processes may exist in an environment. In numerous applications, there is clearly a need for these processes to communicate with each exchanging data or control information. This may be achieved using pipes, signals etc. The parent process spawns K identical child processes along with 2\*K pipes - two for each parent-child pair (one sends messages from parent->child, the other sends messages from child->parent). Two special signals are needed:

● AVAILABLE (30001)

● BUSY (30002)

Initialize: numprime = 0 , primearr[N]

In this assignment what we have to do is :-

1. Parent creates K identical child processes.

2. All child will send **AVAILABLE** to the parent.

3. Parent will provide K random numbers to the child who is **AVAILABLE.**

4. After getting the numbers child will send **BUSY** signal to parent.

5. Each child will check whether there is a prime number in this K random numbers.

1. If YES it will send the numbers to parent. Parent will check whether it has got this prime already from any of the child process.

1. If YES then it will simply discard the number.

2. Else it will insert it into array primearr[] and increase the count numprime++.

1. If numprime == 2\*K then parent will kill all the child process and will report the primes and **return.**

2. Else Parent will send next K random numbers to the child who is **AVAILABLE**. If no available child process it will wait for some child to become available.

2. If NO prime found in the provided K numbers child will send **AVAILABLE** to the parent.

6. Again go to step 3 to 5.

Note: all random integers will be drawn from 1 to 30,000. Example: Say K = 3 count = 0 Parent creates 3 child process C1, C2, C3 Each child sends AVAILABLE to Parent. Parent sends 3 random numbers to each child. To C1 : 31,96,12 To C2: 2,4,6 To C3: 31,2,9 Now all child send BUSY to parent. Next: C1: finds 31 as prime and send it to Parent, Parent increments the count. Now count = 1, after checking with 96 and 12 it sends AVAILABLE C2: checks all the 3 and sends AVAILABLE C3: checks 31 and send it to parent but parent discard it as it already received 31 from C1. Finally it sends AVAILABLE to parent. Whenever parent finds any child AVAILABLE it sends next set of 3 random numbers to it. Thus the step goes on till the total prime count becomes 2\*3 = 6 Note: all the random numbers are drawn from 1 to 50\*3 ie from 1 to 150. Instruction to students: Students will solve this problem taking K = 5 and save the code in \_primepipe.c ps: Give ROLL in all caps without angle bracket.