Date: 1/10/2021

Minor Test-1 [25 Marks]

Q.1. Consider a narrow street which has the direction east-west. It allows cars to go in both directions. As it is a narrow street, the cars must alternate going across the street. It cannot allow more than three cars at a time.

Find a solution to this problem which should not cause the starvation. That is, the cars which want to get across should eventually get across. However, we want to maximize use of the street. Cars should travel across to the maximum capacity of the street (i.e., three cars should go at a time). If a car leaves the street going east and there are no cars from west side, then the next car from east side should be allowed to cross the street. It is not expected a solution, which moves cars across the street three at a time, i.e., east side cars which are waiting should not wait until all three cars that are east side and crossing the bridge have crossed before being permitted to cross. **[10 Marks]**

Q.2

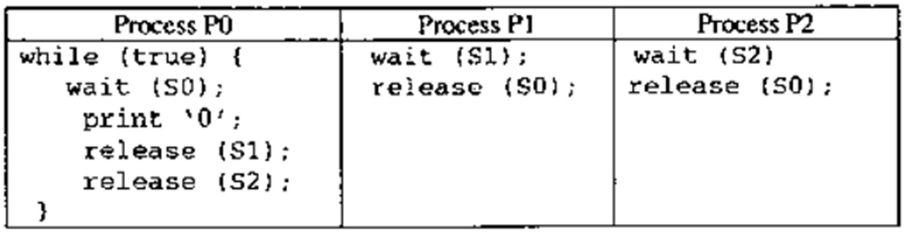
a) What is an output of below given sample program of thread? Explain. **[03 Marks]**

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/types.h>
int main () {
int n, status; pid_t child; n = 0;
if (!(child = fork())) { n = 1; fprintf (stderr, "n = %d\n", n); n = 2;}
else {n = 3; fprintf (stderr, "n = %d\n", n); n = 4; exit (0);
fprintf (stderr, "n =%d\n", n); }
waitpid (child, &status, 0);
n = 5; fprintf (stderr, "n = %d\n", n);
 return 0;
}
```

b) When a thread is created, which resources are used? How these resources different from those when a process is created? **[03 Marks]**

Q.3 In a system, suppose three processes are arriving at time zero and their execution time is 10, 20 and 30 respectively. Consider each process is spending first 20% of execution time to do I/O, next 70% time to do computation, and last 10% time to do I/O again. An OS uses a shortest remaining time first scheduling algorithm and schedules a new process, if the running process gets blocked on I/O or the running process finishes its computation. All I/O operations can be overlapped as much as possible. Derive the percentage of time, a CPU will remain idle? **[06 Marks]**

Q.4 Consider below code of 3 cooperative processes and 3 binary semaphores, which are initialized as S0 = 1, S1 = 0, S2 = 0. How many times will process P0 print ‘0’? Justify your answer. **[03 Marks]**



Process P0	Process P1	Process P2

`while (true) {`
`   wait (S0);`
`    print '0';`
`    release (S1);`
`    release (S2);`
` }`
	`wait (S1);`
	`release (S0);`
		`wait (S2)`
		`release (S0);`