IT3072E Operating Systems, Spring 2022, Midterm exam [1]

Name: _____ Student ID: _____

Question 1: True/False (56pts, 3pts each)

1. Context switch of two user threads is run in the user space: **True**___ **False**___ false
2. The TCB of a user thread is stored in the kernel: **True**___ **False**___ true
3. User thread system calls are initialized in the user space: **True**___ **False**___ true
4. User thread system calls are processed in the user space: **True**___ **False**___ false
5. Only processes can create threads using the command "pthread_create()", this command cannot be used from inside another thread: **True**___ **False**___ false
6. When a process creates a thread using the command "pthread_create()", this command returns the TID of the child thread to the process and return the TID of the parent to the newly created thread: **True**___ **False**___ false
7. In normal circumstances, two different threads cannot share the same runtime stack: **True**___ **False**___ true
8. The exit(0) command executed inside a thread will also cause the parent process to exit: **True**___ **False**___ true
9. The command getpid() returns the PID of the parent process to the child process: **True**___ **False**___ false
10. A thread switches to the kernel stack following a system call: **True**___ **False**___ true
11. A thread in "running" state can transit to 3 different states: waiting, ready and terminated: **True**___ **False**___ false
12. A user thread that runs in kernel mode has the same scheduling priority as kernel threads: **True**___ **False**___ false
13. The FCFS scheduling algorithm optimizes the performance of CPU bound threads: **True**___ **False**___ true
14. We cannot have two different threads passing the name of a same function in the command pthread_create(), thus executing the same function: **True**___ **False**___ false
15. The ready queue stores threads that are not currently running, but are capable of resuming execution: **True**___ **False**___ true
16. Only threads are context-switched, but if the two threads in a context switch do not belong to the same process, then the process is context-switched as well: **True**___ **False**___ true
17. If a parent process creates several processes by executing a sequence of commands fork(), the parent process only remember the PID of the last process in the sequence: **True**___ **False**___ false
18. Processes always have at least one stack because they always have at least one thread: **True**___ **False**___ true

Question 2: Processes (9pts)

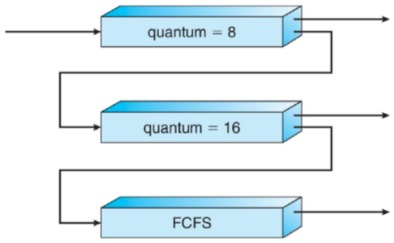
```
pid_t = childpid;
int i;
childpid = fork();
pthread_create();
if (childpid == 0){
   for (i=1; i<2; i++){
      fork();
      pthread_create();
   }
}
fork();
```

In the C code above, not including the original process, how many processes are created (in other words, how many fork() are executed): a) 3___; b) 4___; c) 5___; d) 6___ Solution: c

Question 3: Threads (9pts)

Using the same code as in the previous question, how many threads are created (in other words, how many pthread_create() are executed): a) 2___; b) 3___; c) 4___; d) 5 ___ solution: c

Question 4: Scheduling (20pts)



Assume 4 processes P_0, P_1, P_2, and P_3 arriving in this order at time 0 in the ready queue (quantum = 8) of the multilevel feedback queue of the figure above (4pts each)

1. The CPU burst time of each process is as follow: $P_0 = 16$, $P_1 = 28$, $P_2 = 8$, and $P_3 = 16$. What is the turnaround time of process P_0? a) 24___; b) 16___; c) 40___ solution: c
2. The CPU burst time of each process is as follow: $P_0 = 24$, $P_1 = 28$, $P_2 = 4$, and $P_3 = 20$. What is the waiting time of process P_0 in the second queue? a) 20___; b) 16___; c) 28___ Solution: a
3. The CPU burst time of each process is as follow: $P_0 = 16$, $P_1 = 28$, $P_2 = 20$, and $P_3 = 8$. What is the waiting time of process P_1? a) 48___; b) 44___; c) 24___ Solution: b
4. The CPU burst time of each process is as follow: $P_0 = 16$, $P_1 = 28$, $P_2 = 16$, and $P_3 = 24$. What is the turnaround time of process P_3? a) 84___; b) 92___; c) 88___ Solution: None of them, solution 80. Everyone got 4 points for this question, those who wrote 80 got 4 more points.
5. The CPU burst time of each process is as follow: $P_0 = 24$, $P_1 = 28$, $P_2 = 24$, and $P_3 = 20$. What is the waiting time of process P_3 in the second queue (quantum = 16)? a) 72___; b) 48___; c) 52___ Solution: b

Question 5: Multilevel feedback queues scheduling (8pts)

The table below describes a 5 levels feedback queue similar to Solaris feedback queue for time-sharing and interactive threads, where 0 is the lowest priority. In this table "Time quantum expired" is the new priority of a process that did not complete its current CPU burst in the allocated time quantum of the queue, while "Return from I/O" is the new priority of a process that enters an I/O phase before the end of the allocated time quantum of the queue.

Priority	Quantum time	Time quantum expired	Return from I/O
0	10	0	2
1	8	0	2
2	6	0	3
3	4	1	4
4	2	1	4

We have 3 processes P_0, P_1 and P_2 which have execution cycles as described below. For example, the arrival time of P_0 is 0, its first CPU burst last 5ms, then enters into an I/O queue for 3ms, then a CPU burst of 5ms, an I/O of 4ms, CPU burst of 4ms and then exit.

P_0	CPU	I/O	CPU	I/O	CPU	exit
0	5	3	5	4	4	

P_1	CPU	I/O	CPU	exit
1	8	4	4	

P_2	CPU	I/O	CPU	exit
2	2	6	2	

There is one CPU. The possible **states** of a process are the following: 1) ready to execute: "RY-x" where x is the priority queue in which the process is waiting; 2) running: "RU-x" where x is the priority queue in which the process was before been scheduled to run; 3) waiting in an I/O queue: "I/O-x" where x refers to the priority queue in which the process will be placed after completing its I/O; 4) exited: "EX"

All the processes start in priority queue 2. Arrival times of process P_0 is 0, process P_1 is 1 and process P_2 is 2. The scheduler always empty first the queue of higher priority before it starts to run processes in the next lower priority queue. The scheduler never preempts a currently running lower priority process. The tables below describe 3 possible scheduling for the first 20 ms, which one is correct considering the CPU bursts and I/O bursts of P_0, P_1 and P_2. Solution is a) although there was a small mistake in column 19 for P_0 as one student pointed out

a)

	4	6	14	19
P_0	RU-2	I/O-3	RU-3	I/O-3
P_1	RY-2	RU-2	RY-0	RY-0
P_2	RY-2	RY-2	RY-2	RU-2

b)

	4	6	14	19
P_0	RU-2	I/O-0	RY-0	RU-0
P_1	RY-2	RU-2	I/O-3	RY-3
P_2	RY-2	RY-2	RU-2	RU-2

c)

	4	6	14	19
P_0	RU-2	I/O-3	RU-3	EX
P_1	RY-2	RY-2	RY-0	RY-0
P_2	RY-2	RU-2	RY-3	RU-3