Birzeit University
Department of Electrical and Computer Engineering
ENCS3390 Midterm Exam 1st semester 2021/22

Q1) **True or false, add line of explanation as to WHY provide your final answer in the table below**

1	2	3	4	5	6	7	8	9	10
T	F	F	F	F	F	F	F	T	F
11	12	13	14	15	16	17	18	19	20
F	F	T	F	T	F	F	T	F	T

1) **[T/F]** An operating system is a program that acts as an intermediary between the user of a computer and the computer hardware

   Users cannot access hardware except through the OS

2) **[T/F]** A user-level process can modify its own page table entries

   Page table entries can be modified in kernel mode only, i.e., by the OS

3) **[T/F]** In a multiprocessor system with enough CPUs (cores) a process can get assigned to a given processor (core) permanently to avoid context switches

   No relation. Context switch can happen even if a process is assigned to a given processor (core) permanently

4) **[T/F]** A process can move form a **ready** state to the **waiting** state when I/O is ready

   A process can move form a ready state to the running state, when the CPU scheduler selects it

5) **[T/F]** Shortest Job First scheduling algorithms can never lead to starvation

   SJF can lead to starvation, because short jobs keep coming all the time preventing longer jobs from being scheduled.

6) **[T/F]** A SJF scheduler may preempt a previously running longer job

   Only after finishing previously running job, we invoke the SJF scheduler. SRTF is the preemptive version of SJF

7) **[T/F]** If all jobs have identical run lengths, a RR scheduler (with a time-slice much shorter than the jobs' run lengths) provides better average turnaround time than FIFO (FCFS)

   They will take much longer turnaround time, because of more frequent context switches.

8) **[T/F]** The shorter the time quantum, the more RR scheduler looks like a FIFO (FCFS) scheduler.

The longer the time quantum, the more RR scheduler looks like a FIFO (FCFS) scheduler, because most processes will finish within one time quantum, if not all, i.e., FCFS

9) **[T/F]** Threads that are part of the same process do not share the same stack

Each thread will have its own stack

10) **[T/F]** With kernel-level threads, multiple threads from the same process cannot be scheduled on multiple CPUs simultaneously.

Each process's thread can be mapped to one kernel-level thread, each kernel-level thread can be mapped to a difference CPU, and hence the process's threads are scheduled on multi-CPUs simultaneously.

11) **[T/F]** The degree of multiprogramming is the max number of jobs that can be in the running state at any given time

The degree of multiprogramming is the maximum number of processes that a single-CPU system can accommodate efficiently at any given time

12) **[T/F]** OpenMP provides support for parallel programming in distributed-memory environments

OpenMP provides support for parallel programming in shared-memory environments

13) **[T/F]** The nonzero number returned to the parent process when the system call **fork ()** succeeds is the process ID of the newly created child process.

When the system call fork () succeeds, it returns zero to the child and nonzero to the parent. The nonzero number is the process ID of the newly created child process, and the zero number is the child process ID of the newly created child

14) **[T/F]** If a processor has four cores, and each core has two hardware threads, then the maximum number of software threads that can be run in parallel is four.

This processor has eight hardware threads, and each hardware thread is viewed by the operating system as one logical core. Thus, the maximum number of software threads that can be run in parallel on this processer is eight

15) **[T/F]** A process cannot move from the **waiting** state to the **terminated** state

The process can move from the waiting state to the ready state

16) **[T/F]** In a single CPU system, we can have multiple processes in the **running** state

In a single CPU system, there is at most one process in the running state

17) **[T/F]** Each process has its own inverted page table

A single inverted page table is used to represent the paging information of all processes.

18) **[T/F]** The base (relocation) and the limit registers of a process can be loaded by privileged instructions only

Yes, for protection, these registers can be loaded only in the kernel mode by privileged instructions.

19) **[T/F]** For a program to execute, it should be entirely loaded to memory

For a program to execute, it should be in memory at least partially.

20) **[T/F]** Swapping allows the total physical memory allocated to all processes in a system to exceed the total physical memory size

Yes, because some processes can be swapped temporarily out of memory to a backing store

Birzeit University
Department of Electrical and Computer Engineering
ENCS3390 Midterm Exam 1st semester 2021/22

Q2) Consider the following process arrival, CPU bursts (in seconds) and explicit **priorities** of the processes. Assume that **5** represents a higher priority than 1. For each of the algorithms listed below, draw the Gantt chart. Moreover, calculate the finish time **(F)**, the turnaround time **(TA)**, the waiting time **(W)** of each process for each algorithm. Then, calculate the average of these time quantities for all processes for each algorithm. (Please write your final answers in the specified area)

process	Arrival	CPU Burst	Priority	Priority			SJF			SRTF			RR		
				F	TA	W	F	TA	W	F	TA	W	F	TA	W
A	0	5	2	10	10	5	5	5	0	5	5	0	8	8	3
B	2	3	4	5	3	0	8	6	3	8	6	3	6	4	1
C	4	7	1	17	13	6	17	13	6	17	13	6	17	13	6
D	7	2	3	9	2	0	10	3	1	10	3	1	13	6	4
average				10.25	7	2.75	10	6.75	2.5	10	6.75	2.5	11	7.75	3.5

a) Priority preemptive

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Process	A	A	B	B	B	A	A	D	D	A	C	C	C	C	C	C	C

b) SJF: Shortest Job First

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Process	A	A	A	A	A	B	B	B	D	D	C	C	C	C	C	C	C

c) SRTF: Shortest Remaining Time First

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Process	A	A	A	A	A	B	B	B	D	D	C	C	C	C	C	C	C

d) RR: Round Robin (q=3)

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Process	A	A	A	B	B	B	A	A	C	C	C	D	D	C	C	C	C