Name: Ngô Hồng Quốc Bảo ID: B1809677

OPERATING SYSTEMS (CT104H)

ASSIGNMENT # 3

Please use English to answer the following questions.

Declaration of own work

I, ____*Ngô Hồng Quốc Bảo*____, certify that this assignment is my own work, is not copied from any other person's work.

CHAPTER 3: PROCESSES

1. What are processes?
2. What are the states of a process?
3. Where is the information about the processes stored?
4. What is “context switch cost”?
5. What are the different types of scheduling queues? What are the roles of each scheduling queue?
6. Describe the differences among short-term, long-term and medium-term scheduling?
7. Describe the methods to share resources between a process and its child?
8. How processes exchange information?
9. When does a process terminate?
10. What is the difference between zombie processes and orphan processes?

CHAPTER 4: CPU SCHEDULING

11. What advantage is there in having different time-quantum sizes at different levels of multilevel queueing system?
12. What are the methods to evaluate CPU scheduling algorithms? How to perform these algorithm methods?

CHAPTER 6: DEADLOCKS

13. What are deadlocks?
14. Describe the conditions that are necessary to achieve deadlock ?
15. Describe the methods for handling deadlocks?
16. (Exercise 7.3 – OS Concepts, the 9th Edition, Abraham Silberschatz) Consider the following snapshot of a system

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

Answer the following questions using the banker's algorithm:

a. What is the content of the matrix Need?

b. Is the system in a safe state?

c. If a request from process P1 arrives for (0,4,2,0), can the request be granted immediately?

Task

1. A process can be thought of as a program in execution. A process will need certain resources—such as CPU time, memory, files, and I/O devices—to accomplish its task.
2. States of process:
   - *New*. The process is being created.
   - *Running*. Instructions are being executed.
   - *Waiting*. The process is waiting for some event to occur (such as an I/O completion or reception of a signal).

- *Ready*. The process is waiting to be assigned to a processor.
- *Terminated*. The process has finished execution.

3. Each process is represented in the operating system by a process control block, also called a task control block.
4. Context-switch time is pure overhead, because the system does no useful work while switching. Switching speed varies from machine to machine, depending on the memory speed, the number of registers that must be copied, and the existence of special instructions (such as a single instruction to load or store all registers).
5. Three types of scheduling queues:
   - **Job queue** – It helps you to store all the processes in the system.
   - **Ready queue** – This type of queue helps you to set every process residing in the main memory, which is ready and waiting to execute.
   - **Device queues** – It is a process that is blocked because of the absence of an I/O device.
6. The differences among short-term, long-term and medium-term scheduling:

Long-term	Short-term	Medium-term
Long term is also known as a job scheduler.	Short term is also known as CPU scheduler.	Medium-term is also called swapping scheduler.
It is either absent or minimal in a time-sharing system.	It is insignificant in the time-sharing order.	This scheduler is an element of Time-sharing systems.
Speed is less compared to the short term scheduler.	Speed is the fastest compared to the short-term and medium-term scheduler.	It offers medium speed.
Allow you to select processes from the loads and pool back into the	It only selects processes that is in a ready state of the execution.	It helps you to send process back to memory.

memory		
Offers full control	Offers less control	Reduce the level of multiprogramming.

7. The process' child inherits the parent's process opened handles, files, console input/output, and anonymous pipes.
8. Cooperating processes require an interprocess communication (IPC) mechanism that will allow them to exchange data and information. There are two fundamental models of interprocess communication: shared memory and message passing. In the shared-memory model, a region of memory that is shared by cooperating processes is established. Processes can then exchange information by reading and writing data to the shared region.
9. A process terminates when it finishes executing its final statement .
10. The difference between zombie processes and orphan processes:
    - A **zombie** process is a process that has terminated, but whose parent has not yet called wait(). (The wait() system call is passed a parameter that allows the parent to obtain the exit status of the child. This system call also returns the process identifier of the terminated child so that the parent can tell which of its children has terminated).
    - If a parent did not invoke wait() and instead terminated, thereby leaving its child processes as **orphans**.
11. Processes that need more frequent servicing, for instance, interactive processes such as editors, can be in a queue with a small time quantum.
    Processes with no need for frequent servicing can be in a queue with a larger quantum, requiring fewer context switches to complete the processing, and thus making more efficient use of the computer.
12. There are two methods to evaluate CPU scheduling algorithms:
    - **Preemptive Scheduling:** In Preemptive Scheduling, the tasks are mostly assigned with their priorities. Sometimes it is important to run a task with a higher priority before another lower priority task, even if the lower

priority task is still running. The lower priority task holds for some time and resumes when the higher priority task finishes its execution.

- **Non-Preemptive Scheduling:** In this type of scheduling method, the CPU has been allocated to a specific process. The process that keeps the CPU busy will release the CPU either by switching context or terminating. It is the only method that can be used for various hardware platforms. That's because it doesn't need special hardware (for example, a timer) like preemptive scheduling.

13. The situation when a waiting process is never again able to change state, because the resources it has requested are held by other waiting processes is called deadlocks.

14. A deadlock situation can arise if the following four conditions hold simultaneously in a system:

- Mutual exclusion. At least one resource must be held in a nonsharable mode; that is, only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.
- Hold and wait. A process must be holding at least one resource and waiting to acquire additional resources that are currently being held by other processes.
- No preemption. Resources cannot be preempted; that is, a resource can be released only voluntarily by the process holding it, after that process has completed its task.
- Circular wait. A set {P0, P1, ..., Pn} of waiting processes must exist such that P0 is waiting for a resource held by P1, P1 is waiting for a resource held by P2, ..., Pn-1 is waiting for a resource held by Pn, and Pn is waiting for a resource held by P0.

15. There are three ways to deal with the deadlock problem:

- Deadlock prevention or avoidance - Do not allow the system to get into a deadlocked state.
- Deadlock detection and recovery - Abort a process or preempt some resources when deadlocks are detected.

- Ignore the problem all together - If deadlocks only occur once a year or so, it may be better to simply let them happen and reboot as necessary than to incur the constant overhead and system performance penalties associated with deadlock prevention or detection.

16. 4 resources type: A (3 instances), B (14 instances), C (12 instances) and D (12 instances).

a) Matrix **Need**:

	A	B	C	D
P0	0	0	0	0
P1	0	7	5	0
P2	1	0	0	2
P3	0	0	2	0
P4	0	6	4	2

b) Matrix **Work**:

	A	B	C	D
	1	5	2	0
P0	1	5	3	2
P3	1	11	6	4
P4	1	11	7	8
P2	2	14	12	12
P1	3	14	12	12

The system is in safe state since the sequence < P0, P3, P4, P2, P1 > satisfies safety criteria.

c) The request can be granted immediately.

- New matrix Allocation:

	A	B	C	D
P0	0	0	1	2
P1	1	4	2	0
P2	1	3	5	4
P3	0	6	3	2
P4	0	0	1	4

- New matrix Available:

A	B	C	D
1	1	0	0

- New matrix Need:

	A	B	C	D
P0	0	0	0	0
P1	0	3	3	0
P2	1	0	0	2
P3	0	0	2	0
P4	0	6	4	2

- New Matrix Work:

	A	B	C	D
	1	1	0	0

P0	1	1	1	2
P3	2	4	6	6
P4	3	8	8	6
P2	3	14	11	8
P1	3	14	12	12