Part 1: (80 points - 4 points for each problem)

( b ) 1. Which is the advantage of multiprocessor systems?
   (a) Increased modularity (b) Increased reliability (c) Increased security (d) None of above

( c ) 2. The scheduler that brings processes into memory and swaps them out on disk as needed is referred to as:
   (a) Short-term scheduler (b) Long-term scheduler (c) Medium-term scheduler (d) None of the above

( b ) 3. The purpose of process synchronization is?
   (a) Let users run different processes independently
   (b) Avoid races with respect to access to a shared resource
   (c) Ensure progress where processes do not get into deadly embraces
   (d) None of above

( a ) 4. CPU Scheduling algorithms are used for:
   (a) Choosing the next process to run from the PCBs in the ready list
   (b) Putting to sleep and waking up processes in an efficient manner
   (c) Allocating memory to the processes in a fair and efficient way
   (d) None of the above

( d ) 5. Which is not the property of a signal in UNIX systems?
   (a) A signal is generated by the occurrence of a particular event.
   (b) A generated signal is delivered to a process.
   (c) Once delivered, the signal must be handled.
   (d) None of the above.

( c ) 6. Which is not a primary reason for providing an environment that allows process cooperation:
   (a) Modularity (b) Computational speedup (c) Concurrency (d) Information sharing

( d ) 7. CPU burst distribution is generally characterized as
   (a) Constant (b) Linear (c) Polynomial (d) Exponential or hyper-exponential

( a ) 8. Which is a hardware solution to the critical-section problem?
   (a) TestAndSet (b) Shared memory (c) Semaphore (d) Monitor

( d ) 9. Which is not the function of a dispatcher?
   (a) Switching context
   (b) Switching to use mode
   (c) Jumping to the proper location in the user program to restart that program
   (d) None of above

( b ) 10. Which is not a CPU scheduling criterion?
     (a) CPU utilization (b) Burst time (c) Throughput (d) Response time

( b ) 11. Which is a non-preemptive scheduling?
     (a) RR (b) FCFS (c) SRTF (d) None of the above

( d ) 12. Shortest Remaining Time First scheduling is a special case of:
     (a) Preemptive priority scheduling (b) Non-preemptive priority scheduling
     (c) Optimal scheduling algorithm (d) None of the above

( b ) 13. Which of the memory allocation schemes are subject to external fragmentation?
     (a) Multiple Contiguous Fixed Partitions (b) Segmentation (c) Paging (d) None of above
14. A computer provides the user with virtual address space of $2^{32}$ words. Pages of size 4096 ($2^{12}$) words. If the hexadecimal virtual address is 23456789, the page number in hexadecimal would be:
   (a) 234567 (b) 56789 (c) 456789 (d) 34567 (e) 23456

15. In demand-paged system where page size is 2048 words, and the available physical memory is equal to $2^{17} = 128$ K words, the length of the physical address is equal to:
   (a) 16 bits (b) 28 bits (c) 17 bits (d) None of the above

16. The type of paging where a page is never rolled (or swapped) into memory unless needed is referred to as:
   (a) Daemonized lazy paging (b) Lazy paging daemon (c) Lazy swapper (d) Demand paging

17. The dirty bit is used for the purpose of:
   (a) Dynamic allocation of memory used by one process to another
   (b) Implementing FIFO page replacement algorithm
   (c) To reduce the average time required to service page faults
   (d) None of the above

18. Which page-replacement algorithm suffers from Belady’s anomaly?
   (a) Least recently used (LRU) (b) First in first out (FIFO)
   (c) Not recently used (NRU) (d) None of the above

19. Working set model is:
   (a) Used for finding the minimum number of frames necessary for a job, so that jobs can run without "thrashing"
   (b) Used to find out the average number of frames a job will need in order to run smoothly without causing thrashing
   (c) Used to determine whether page replacement is needed
   (d) All of the above.

20. Which is not a file attribute?
   (a) Name (b) Size (c) Shape (d) Type

Part 2: (120 points)

1. Describe the process status.
   • new: The process is being created.
   • running: Instructions are being executed.
   • waiting: The process is waiting for some event to occur.
   • ready: The process is waiting to be assigned to a process.
   • terminated: The process has finished execution.

2. Short Answer:
   (a) What are main differences between user-level threads and kernel-level threads?
   (b) Give one situation where user-level thread is preferred.
   (c) Give one situation where kernel-level thread is preferred.

   (a) User-level threads are unknown by the kernel, whereas the kernel is aware of kernel threads.
   • User threads are scheduled by the thread library and the kernel schedules kernel threads.
   • Kernel threads need not be associated with a process whereas every user thread belongs to a process.

   (b) Thread creations are less costly.
   • The thread-scheduling can be customized or changed to suit specific applications.
The threads can take advantage of a multiprocessor.
- A thread that takes a page fault won’t block the entire process.
- Threads within different processes can be scheduled according to a single scheme of relative prioritization.

3. How many processes will be created when the following program is executed?
Assume that all fork system calls are successful. What will be printed?
(Hint: Be careful and draw a picture.)

```c
main()
{
    int i=1;
    int ret_val;

    while(i <= 3)
    {
        if ((ret_val = fork()) == 0) { /* Child’s code */
            printf("In child %d. \n", i);
            i = i + 1;
        } else { /* Parent’s code */
            printf("In parent %d. \n", i);
            exit(0);
        }
    }
}
```

There are 4 processes (1 parent and 3 child processes) created when this program is executed. The following could be printed:

- In parent 1.
- In child 1.
- In parent 2.
- In child 2.
- In parent 3.
- In child 3.

4. Here is the weak reader preference solution to the readers writers problem using semaphores:

```c
semaphore db, mutex = 1,1;
shared int reader_count = 0;

Reader: Writer:

DOWN(mutex);
reader_count++;
if (reader_count==1) DOWN(db);
UP(mutex);

DOWN(db);
- write to database -
UP(db);

DOWN(mutex);
reader_count--;
if (reader_count==0) UP(db);
UP(mutex);
```

Rewrite the above solution using message passing instead of semaphores; in particular use mailboxes (UNIX pipes) and the notation write(mbox,msg) and read(mbox,msg).
msg = 0;

Reader:
read(mutex,msg);
reader_count++;
if (reader_count==1) read(db, msg);
write(mutex);
- read from database -
read(mutex,msg);
reader_count--;
if (reader_count==0) write(db, msg);
write(mutex,msg);

Writer:
read(db,msg);
- write to database -
write(mutex);

5. Suppose that the following processes arrive for execution at time 0 in the order A, B, C:

<table>
<thead>
<tr>
<th>Process</th>
<th>Run Time</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2</td>
<td>1=high</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>3=low</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

(a) Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, a nonpreemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1) scheduling.

(b) What is the waiting time of each process for each of the scheduling algorithms?

(c) What is the turnaround time of each process for each of the scheduling algorithms?

6. P is a set of processes. R is a set of resources. E is a set of request or assignment edges. The sets P, R, and E are as follows:

\[
P = \{P_1, P_2, P_3\} \quad R = \{R_1, R_2, R_3\} \\
E = \{P_1 \rightarrow R_1, P_2 \rightarrow R_3, P_3 \rightarrow R_2, R_1 \rightarrow P_2, R_2 \rightarrow P_2, R_2 \rightarrow P_1, R_3 \rightarrow P_3\}
\]

- \(R_1\) has one instance.
- \(R_2\) has two instances.
- \(R_3\) has one instance.

(a) Draw the resource-allocation graph.

(b) Is there any deadlock in this situation? Briefly Explain.

(b) Consider the resource-allocation graph. One cycle exist in the system.

\(P_1 \rightarrow R_1 \rightarrow P_2 \rightarrow R_3 \rightarrow P_3 \rightarrow R_2 \rightarrow P_1\)

P1, P2, and P3 are deadlocked.

7. Given memory partitions of 100 KB, 500 KB, 200 KB, 300 KB, and 600 KB (in order).

(a) How would each of the first-fit, best-fit, and worst-fit algorithms place processes of 212 KB, 417 KB, 112 KB, and 426 KB (in order)?

(b) Which algorithm makes the most efficient use of memory?

- First fit: (a) 500 KB (b) 600 KB (c) leftover of 600 KB (d) None
- Best fit: (a) 300 KB (b) 500 KB (c) 200 KB (d) 600 KB
- Worst fit: (a) 600 KB (b) 500 KB (c) 300 KB (d) None

8. Consider a paging system where the page table is stored in cache memory. The hit ratio is 80% meaning the page table entry will be found in cache 80% of the time. The normal memory access time is \(t = 100\) nanoseconds.

(a) If checking the entry in the cache takes 20 nanoseconds, what is the effective (average) paged memory access time?

(b) What hit ratio is needed to make the effective access time to 130 nanoseconds?
1. $0.8 \times (100 + 20) + 0.2 \times (100 \times 2 + 20)$
   
   $= 0.8 \times 120 + 0.2 \times 220 = 140.$

2. $120 \ p + 220 \times (1 - p) = 130$
   
   $120 \ p + 220 - 220 \ p = 130$
   
   $100 \ p = 90$
   
   $p = 90\%$

9. What are three requirements (criteria, properties) of good solution to the critical-section problem?

   (a) Mutual exclusion is guaranteed.

   (b) Progress is maintained. Processes not in their critical sections may not block other processes from entering their critical sections.

   (c) Bounded waiting is assured.
      
   i. There is no starvation; that is, a process wanting to enter its critical section is allowed to do so in finite time.

   ii. No assumptions are made about the relative speeds of processes or the number of processors (CPUs).

10. Suppose there are 16 virtual pages and 4 page frames. Determine the number of page faults that will occur with the reference string 0 1 7 2 3 2 7 1 0 3 12 13 12 7 1, if the page frames are initially empty, using each of the following page replacement algorithms: (a) FIFO (b) LRU (c) Optimal.

   FIFO: 10
   
   LRU: 11
   
   Optimal: 8