Operating Systems - 1st Term - 4th class

Q: 1: A: Advantages of multithreaded programming:

1. Responsiveness: allow program to continue running even if part of it is blocked.

2. Resource sharing: sharing code and data will allow an application to have several different threads within the same address space.

3. Economy: Allocating memory and resources of process creation is costly, because threads share resources it is more economical to create and context-switch threads, so it is much more time consuming to create and manage processes than threads.

4. Utilization of multiprocessor architectures: the benefits of multithreading can be greatly increased in a multiprocessor architecture.

B. Page Replacement Approach:

If no frame in memory is free, we find one that is not currently being used and free it, then use this free frame to load a new frame from disk.

Its algorithms include:

1. FIFO Page Replacement
2. Optimal page Replacement
3. LRU Page Replacement
Q2: A: Mechanisms of time-sharing operating systems
a. CPU scheduling and multiprogramming to provide each user with a small portion of time-shared computer.
b. Job scheduling: when there is no enough memory then the system must choose a job from disk to load it into memory.
c. Virtual memory and memory management system.
   A technique that allows the execution of a process that is not completely in memory so it will enable users to run programs that are larger than actual physical memory.

\[
\begin{array}{c|c|c|c|c}
P_2 & P_3 & P_4 & P_5 \\
0 & 3 & 11 & 22 & 31
\end{array}
\]

b. \( P_1 \) waiting = 3
   \( P_2 \) waiting = 0
   \( P_3 \) waiting = 11
   \( P_4 \) waiting = 22

c. Average waiting time = \( \frac{0 + 3 + 11 + 22}{4} = \frac{36}{4} = 9 \) msec.
Q3: Storage management:

1. File system management:
   - Creating and deleting files
   - Creating and deleting directories
   - Supporting primitives for manipulating files and directories
   - Mapping files on secondary storage
   - Backing up files on stable storage media

2. Mass storage management:
   - Free space management
   - Storage allocation
   - Disk scheduling

3. Caching

4. IO systems:
   - Memory management
     - A general device-driver interface
     - Drivers for specific hardware devices

5. Message-passing system:
   - Provides a mechanism to allow processes to communicate and to synchronize their action without sharing the same address space and is useful in distributed environment.
   - If processes P & Q want to communicate, they must send messages to and receive messages from each other through communication link, which can be:
     - Direct or indirect communication
     - Synchronous or asynchronous communication
     - Automatic or explicit buffering
4.1. Operating System Problems:

1. Process Synchronization with Critical-section problem:

Consider a system consisting of \( n \) processes \( \{P_0, P_1, P_2, \ldots, P_n\} \). Each process has a segment of code, called critical-section, in which the process may be changing common variables, updating a table, writing a file. The important feature of the system is that when one process is executing in its critical section, no other process is to be allowed to execute in its critical section.

Solution must satisfy the following requirements:

1. Mutual Exclusion
2. Progress
3. Bounded waiting

The methods that are used include:

1. Peterson's Solution
2. Synchronization Hardware
3. Semaphores
2. Deadlock: A set of processes with a set of resources each process in this set is holding a resource that need it to be executed and at the same time is waiting another resource which is held by another process, so the deadlock condition include:

1. Mutual Exclusion
2. Hold & wait
3. No Preemptive
4. Circular wait

The solution to deadlock problem include:
1. Deadlock Prevention: by preventing one of the condition of deadlock
2. Deadlock Avoidance: by checking the safe and unsafe state of the processes and resources by using Safety Algorithm and Banker's Algorithm
3. Recovery from deadlock and detecting deadlock by process Termination or Resource Preemption.
Advantages of paging mechanism:

1. Paging permits the physical address space of a process to be noncontiguous, and it avoids the problem of fitting memory chunks of varying sizes onto the backing store.

2. Protection: when building the page table for the process, an additional field is used to set a valid/invalid bit to the page to indicate that it is protected or not.

3. Shared pages: means that when two processes have the same pages in memory, there is no need to load these shared pages twice in memory; instead, it can be referred to it only by the page table for each one.
### OS: B: UNIX system structure:

<table>
<thead>
<tr>
<th>the (users)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Shells and commands</td>
<td></td>
</tr>
<tr>
<td>Compilers and interpreters</td>
<td></td>
</tr>
<tr>
<td>System libraries</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System-call interface to the kernel</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Signals</td>
<td>Terminal handling</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Kernel interface to the hardware</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Terminal controllers &amp; terminals</td>
<td>Device controllers</td>
</tr>
</tbody>
</table>

Figure: Unix system structure
Q6:
(a) Matrix Need:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₀</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>P₁</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>P₂</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>P₃</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>P₄</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

(b) Safe state: no, the safe sequence is:

\[ \langle P₁, P₃, P₄, P₂, P₀ \rangle \]

(c) To decide whether this request can be granted immediately, if Request ≤ Available

\[ (1,0,2) ≤ (3,3,2) \Rightarrow True \]

(Yes), so it is true, therefore the request \( (1,0,2) \) can be granted immediately.

\[ \underline{Finish} \]

Dr. Reem J. E.