Date: / /

Final Term

Name :- Danial Rahman
ID # :- 15433
Subject :- Opreating System
Degree :- BSSE
Semester :- 3rd
Teacher :- M. Daud Khan

Q no3 :-

Ans :- The most Suitable Component of an opreating System that Suited to ensure fair, Secure, orderly and efficient use of memory is Memory Management System.
The task of memory management includes keeping track of used and free memory Space as well as when where and how much memory to allocate and deallocate.
it is also responsible for Swapping process in and out of Main memory. The purpose of a memory Management is to ensure fair Secure orderly and efficient use of a memory.

Date: / /

Q no:-

Ans:-

Dynamic loading:- Dynamic loading is the process of loading the dependent library or routine on demand or at some time at run time, load time (The time which the many program executable is loaded into memory. Contrast to loading all dependencies at load time together with the main program. The window loader for example support both techniques. The loading process complete when the library has been successfully loaded into main memory.

Examples:- when we write a program C language we are going to execute the same program

```
if { }
move ( )
```

Date: / /

Dynamic Linking:

System library or other routine is loaded during run-time and its supported by OS. When the program attempt to call an imported function for the first time, that library that contains that function they may not have been loaded yet, initially the compiler place a temporary small function called a Stub that gets called instead of the important function The Stubs call into the OS if the library is currently not loaded it get loaded (this step is called dynamic loading) Then the Stub is modified so that it call the important function directly the next time it gets called. This process is called dynamic linking

In Dynamic linking when a module need to be called that module is loaded into memory and a link b/w the calling module is established by the Stub which is the peace of the code that is linked in linking time program

Qno6:-

Ans:- First-Fit:-

Allocate into the first available gap found of adequate size, Starting from the beginning of memory region list Scan memory from Short for First-Fit must always Ship over many regions at the Start of list

② Next-Fit:-

Allocate into the First available gap-found remaining the Search from the last allocation

Scan memory region list from the point of last allocation to next fit Breaks up large block at the end of memory

③ Best-Fit

Pick the closest free region in the entire list leaves Small unusable region and slower due to Searching of entire list.

(4) worst-Fit

the worst-fit in the entire list leavel ~~small~~ slower as it searched entire list fragmentation still an issue.

Qnos:-

Ans:- Internal fragmentation

Memory block assigned to process is bigger some portion of memory is left unused as it cannot be used by another process

External

Total memory space is enough to satisfy a request or to reside a process in it but it is not contiguous. so it cannot be used External fragmentation can be reduced by compaction or ~~&~~ shuffle ~~co~~ memory content

Internal	External
(1) In Internal fragmentation fixed sized memory block square measure appointed to process	(1) In external fragmentation variable-sized memory block square appointed to method.

Date: / /

Day

② Internal Fragmentation happen when the method or process is larger than the memory	② External Fragmentation happen when the method or process is removed
③ The Solution of internal Fragmentation is best fit block	③ Solution of external is Compaction, Paging and Segmentation
④ Internal Fragmentation occur when memory is devided into fixed size partition	④ When memory is devided into variable size partition based on the size of process
⑤ The difference b/w memory allocate and refused space	⑤ The unused space formed b/w non-contiguous memory.

Qno 1:-

Ans. Yes it is necessary to check that either Safe State exist or not in deadlock prevention because when we disallow deadlock by setting "Safe States", it mean process completion is always ~~guest~~ guaranteed.

Qno 4:-

~~Symmetric~~ Difference B/w Symmetric and Assymetric

- Symetric encryption use a single key that need to be shared among the people who need to recieve the message while assymmetrical encryption uses a pair of public key and a private key to encrypt and decrypt message when communicating
- Symmetric encryption is an old technique while Assymmetric encryption is relatively new.
- Assymmetric encryption was introduced to complement the inherent problem of the need to share the key in Symmetrical encryption model, eliminating the need to share the key by using pair public private key

Assymmetric encryption takes relatively more time than Symmetric encryption

Q.no 7:-

Ans:-

Threads:-

A thread is a Sequential execution Stream within a process. This mean that Single process may be broken up into multiple threads. Each thread has its own program Counter register, and Stacks but they all Share the Same address Space within the process

Kernal Threads - User - level:

Thread can either be Created as Kernel threads Or user level threads

Date: / /

depending on the opreating System In System that use kernel-level-threads the OS itself is aware of each individual thread, & Content Switch b/w kernel thread belonging to same process require only register Program Counter and Stack to be changed the overall memory management information doesn't need to be switched since both of the thread share same address Space. Thus Content switching b/w two kernel threads is slightly faster than Switching b/w two processes

A user level thread is the thread ~~within~~ within a process which the OS does not know about In user level threads approch the cost of a Content Switch b/w threads can be made even lower. Since the OS itself doesn't need to be involved no extra System calls are required