**ASSIGNMENT-1**

**OPERATING SYSTEM**

**ANSWER 1-**

1. An Operating System has many different functions but the main function is to provide interface between a user and hardware.

USER

HARDWARE

OS

If a user wants to directly interact with hardware then for executing small codes one will have to write lengthy codes. So the OS is introduced as an interface between the user and hardware which has System Calls like write(), fork(), kill() etc that will reduce the code length.

1. **Resource Management:**

The Operating System manages all resources that are connected to the system like memory, processors and all the input output devices. The OS will identify at which time the CPU will perform which operation, in which time the memory is used by a program and when the I/O devices are used by which program.

1. **Memory Management:**

The OS is used to provide memory to the process and to deallocate memory form the process.

It also insures that when a process gets completed, the memory of the processes is deallocated.

1. **Process Management:**

It means all the processes that are given by the user or the processes that are system’s own process are handled by OS.

The OS can set priority among processes, can start and stop a process and can create new child process by dividing the large process.

1. **File Management:**

All the operations those are responsible for storing and accessing the files is determine by OS.

**ANSWER 2:**

When CPU has no program to execute it simply run the idle task code. It may execute a special CPU instruction to power down the CPU until a hardware interrupt arrives. ON x86 CPUs it is hlt (halt).

**ANSWER 3:**

int main() // CPU execution

{

Int I,j; // CPU execution

Scanf(“%d”,&i); // I/P execution

for(j=0;j<I;j++) // CPU execution

{

Sum=j+I; // CPU execution

}

printf(“%d”,Sum); // O/P execution

exit(0); // CPU execution

}

**ANSWER 4:**

1. **Multi-Programming:**

J1 J2 J3

CPU

Step1: J1

Step 2: J1 gone for I/O

Step 3: J2 job gets some CPU time

In multi programming more than 1 process resides in the system. Suppose we have three jobs J1, J2 and J3 and J1 get some CPU time and after executing for a while if J1 terminates or if it goes for some I/O at that time job J2 will b given the CPU because of this the CPU will not remain idle, System utilization increases and more process can be executed in fixed time.

1. **Multi-tasking OS:**

J1 J2 J3

CPU

Step1: J1 gets some CPU time.

Step 3: J2 gets some CPU time

Step3

CPU will be multiplexing all the jobs without completing anything. A time slice is given to all the jobs J1, J2 and J3. First J1 will be given CPU for a particular time slice and when the time slice is over job J2 is given CPU for that particular time slice and similarly job J3. After J3 job J1 will be given CPU and this cycle continues till all the jobs are completed. This improves interactiveness and CPU will not remain idle. Preemption is also present in this OS.

1. **Multi-processing OS:**

CPU

CPU

J1

J2

Lot of CPUs are present in the same computer. Lot of jobs are executed parallel.

In a single system if you have many CPU attached then it is a multiprocessing environment. It is faster and more reliable and improves the throughput.

**ANSWER 5:**

A program is a set of instructions that are to perform a designated task, where as the process is an operation which takes the given instructions and perform the manipulations as per the code, called ‘execution of instructions’. A process is entirely dependent of a ‘program’.  
 A process is a module that executes modules concurrently. They are separate loadable modules. Whereas the program perform the tasks directly relating to an operation of a user like word processing, executing presentation software etc.

**ANSWER 6:**

**New State:** Secondary memory.

**Ready State:** Main memory.

**Run State:** Main memory.

**Block State:** Main memory.

**Terminate State:** Main memory.

**Suspend Ready State:** Main memory.

**Suspend Block State:** Secondary memory.

**ANSWER 7:**

If two processes P1 and P2 are present and at any time process P1 is removed from memory and process P2 is assigned the memory.This switching is termed as context switching.

|  |  |  |
| --- | --- | --- |
| Process | Burst Time | Arrival Time |
| P1 | 10 | 0 |
| P2 | 20 | 2 |

**Eg-**

P1 P2

10

30

0

0

So Context switch including time at zero and end time is 3, and Context switch excluding time at zero and end time is 1.

**ANSWER 8:**

|  |  |  |
| --- | --- | --- |
|  | **Minimum** | **Maximum** |
| Ready | 0 | M |
| Running | 0 | N |
| Block | 0 | M |

**ANSWER 9:**

1. **Long Term Scheduler:**

It is also called a job scheduler. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.

It controls degree of Multi-programming. Degree of Multi-programming means number of processes residing in ready queue.

1. **Medium Term Scheduler:**

Medium-term scheduling is a part of swapping. It removes the processes from the memory. It reduces the degree of multiprogramming. The medium-term scheduler is in-charge of handling the swapped out-processes.

1. **Short Term Scheduler:**

It is also called as CPU scheduler. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them. Short-term schedulers, also known as dispatchers, make the decision of which process to execute next.

**ANSWER 10:**

1. **LEARNABILITY:**  
    Another very important core principle is the ability to easily learn and use an interface after using it for the first time.
2. **CONSISTENCY:**  
   As well as **matching people’s expectations** through terminology, layout and interactions the way in which they are used should be consistent throughout the process and between related applications. By maintaining consistency users learn more quickly, this can be achieved by re-applying in one part of the application their prior experiences from another.
3. **MATCH USER EXPERIENCE AND EXPECTATION:**  
   By matching the sequence of steps, layout of information and terminology used with the expectations and prior experiences of the user, the friction and discomfort of learning a new system will be reduced.

**ANSWER 11:**

Virus programs out there in the Internet are not that smart to get into a computer hardware. A CPU is a hardware component and that cannot be attacked directly by a software program (Virus). So the Virus program first needs to get into the Motherboard firmware (BIOS) to take control of the whole system. But currently there are no malwares known to the world which could do such complicated things. It doesn’t mean that they do not exist. A CPU is at the lowest end of all computer components. It is coded in a different language then your operating system (CPU is coded in Assembly and OS mainly in C++, the virus has to take over loads of parts before it can reach the CPU. Also a CPU isn't a single component. It has two more units inside it called the Arithmetic Logic Unit and the Control Unit. It is basically super complex to code a virus to do so.

**ANSWER 12:**

 It is the module that gives control of the CPU to the process selected by the short-term scheduler. It receives control in kernel mode as the result of an interrupt or system call. The functions of a dispatcher involve the following:

* [Context switches](https://en.wikipedia.org/wiki/Context_switch), in which the dispatcher saves the [state](https://en.wikipedia.org/wiki/State_(computer_science)) (also known as [context](https://en.wikipedia.org/wiki/Context_(computing))) of the [process](https://en.wikipedia.org/wiki/Process_(computing)) or [thread](https://en.wikipedia.org/wiki/Thread_(computing)) that was previously running; the dispatcher then loads the initial or previously saved state of the new process.
* Switching to user mode.
* Jumping to the proper location in the user program to restart that program indicated by its new state.

The dispatcher should be as fast as possible, since it is invoked during every process switch. During the context switches, the processor is virtually idle for a fraction of time, thus unnecessary context switches should be avoided.

**ANSWER 13:**

Applications of Real Time operating system:

* Almost all the modern telecommunication systems make use of RTOS.
* Radar systems, network switching control systems, satellite monitoring systems, satellite launch-control and maneuvering mechanisms, global positioning systems all have their roots in RTOS.
* Now a days RTOs are increasingly finding use in strategic and military operations. These are used in guided missile launching units, track-and-trace spy satellites, etc.

**ANSWER 14:**

It provides an interface to the service available by an OS. These calls are generally available as routines written in C and C++.

**ANSWER 15:**

1. fork():

The purpose of fork() is to create a new process, which becomes the child process of the caller. After a new child process is created, both processes will execute the next instruction following the fork()system call. Therefore, we have to distinguish the parent from the child.

1. exec():

In computing, exec is a functionality of an operating system that runs an executable file in the context of an already existing process, replacing the previous executable. This act is also referred to as an overlay.