MultiProcessing

1. Where are the function arguments and variables stored?

Ans:  The local variables and the parameters are always stored in **stack segment** whereas the references are created at heap segments.

1. Where are global variables stored?

Ans: Global variables are stored **in the data section**. Unlike the stack, the data region does not grow or shrink — storage space for globals persists for the entire run of the program. Finally, the heap portion of memory is the part of a program's address space associated with dynamic memory allocation.

1. What are the resources assigned to a process?

Ans: The process needs certain resources such as **CPU and memory** to perform the tasks. Now we will look into the related commands and system calls to know the information on resource utilization and monitoring

1. How are processes identified?

Ans: **Each process running in Windows is assigned a unique decimal number** called the process ID (PID). This number is used in a number of ways, for example to specify the process when attaching a debugger to it.

1. Who selects the process for execution?

Ans: **CPU scheduler** selects a process among the processes that are ready to execute and allocates CPU to one of them

1. What are the guiding principles used by scheduler to select a process?

* Ans: Arrival Time. Arrival time is defined as the point of time at which a process enters into the ready state and ready for its execution.
* Burst Time. ...
* CPU Utilization. ...
* Throughput. ...
* Turnaround Time. ...
* Waiting Time. ...
* Response Time.

1. List atleast 5 scheduling algorithms

Ans: First Come First Serve (FCFS), 2) Shortest-Job-First (SJF) Scheduling, 3) Shortest Remaining Time, 4) Priority Scheduling, 5) Round Robin Scheduling, 6) Multilevel Queue Scheduling.

1. What do you mean by single and multi core?

Ans: A processor that has more than one core is called Multicore Processor while one with single core is called Unicore Processor or Uniprocessor.

1. How many processes can a N core CPU run parallely?

Ans: A single processor can run only **one instruction at a time**: it is impossible to run more programs at the same time. A program might need some resource, such as an input device, which has a large delay, or a program might start some slow operation, such as sending output to a printer.

1. How is a program executed internally? What are the steps involved?

Ans: The following steps are involved in the execution of a program: - Fetch: The control unit is given an instruction. - Decode: The control unit then decodes the newly received instruction. - Execute: During the execution the Control unit first commands the correct part of hardware to take action.

1. What are the various attributes of a process? Mention atleast one command to view process attributes

* Ans: Process ID. When a process is created, a unique id is assigned to the process which is used for unique identification of the process in the system.
* Program counter. ...
* Process State. ...
* Priority. ...
* General Purpose Registers. ...
* List of open files. ...
* List of open devices.

**ps command** is used to show some attributes of a process. This command reads through the kernel's data structures and process tables to fetch the characteristics of a process. By default, ps command displays the processes owned by the user running the command.

1. What are the different states of a process?

Ans: Running − In this state the instructions are being executed. Waiting − The process is in waiting state until an event occurs like I/O operation completion or receiving a signal. Ready − The process is waiting to be assigned to a processor. Terminated − the process has finished execution

1. How do we run multiple processes using a single CPU?

Ans: In case of multi-processor or multi-core environment, each processor/core can be used to run a different process and thus achieving parallelism in the system. While in case of single-processor or single-core, **scheduling algorithms are used to execute processes concurrently**.

1. What do you mean context switch? When does it happen?

Ans: Context Switching **involves storing the context or state of a process so that it can be reloaded when required and execution can be resumed from the same point as earlier**. This is a feature of a multitasking operating system and allows a single CPU to be shared by multiple processes.

1. What does the term concurrency and parallelism mean?

Ans: Concurrency is the task of running and managing the multiple computations at the same time. While parallelism is the task of running multiple computations simultaneously.

1. Why do we need to assign priorities to processes?

Ans: Every process requires a certain amount of system resources, like central processing unit (CPU) time and random access memory (RAM), to be able to perform its tasks. Each process is assigned a process priority, which **determines how much CPU or processor time is allocated to it for execution**

1. Which command is used to view process status in realtime?

Ans: Any time the system is running, processes are also running. You can use the ps command to find out which processes are running and display information about those processes.

1. Which command is used to view process tree with pid details?

Ans: **Pstree command** in Linux that shows the running processes as a tree which is a more convenient way to display the processes hierarchy and makes the output more visually appealing. The root of the tree is either init or the process with the given pid.

1. Which command is used to get pid, ppid and process group id?

Ans: **ps command** is used to list the currently running processes and their PIDs along with some other information depends on different options

1. Which process starts all processes in the system?

Ans: **Whenever a command is issued in Unix/Linux, it creates/starts a new process**. For example, pwd when issued which is used to list the current directory location the user is in, a process starts. Through a 5 digit ID number Unix/Linux keeps an account of the processes, this number is call process ID or PID.

1. How to create a new process from within a program?

Ans:  **Fork system call is used for creating a new process**, which is called child process, which runs concurrently with the process that makes the fork() call (parent process). After a new child process is created, both processes will execute the next instruction following the fork() system call.

1. Where the process information maintained? What is the name of the data structure used to hold process information?

Ans: A **process control block (PCB)** is a data structure used by computer operating systems to store all the information about a process.

The **doubly-linked list** data-structure is used 'generally' to implement Process Control Block.

1. What happens on exit()?

Ans: **Calling the exit subroutine terminates the entire process, including all its threads**. In a multithreaded program, the exit subroutine should only be used when the entire process needs to be terminated

1. What is the difference between exit() and \_exit()? Which will cause quick exit?

Ans: **\_exit() won't flushes the stdio buffer while exit() flushes the stdio buffer prior to exit**. \_exit() can not perform clean-up process while exit() can be registered with some function ( i.e on\_exit or at\_exit) to perform some clean-up process if anything is required before existing the program.

1. Does \_exit close open fds?

Ans: **Does guarantee that all open files will be closed if your program terminates normally** (i.e. via exit or a return from main ). However, if your program terminates abnormally, e.g. it's closed by the operating system due to using a NULL pointer, it's up to the operating system to close the files.

1. Does \_exit flush open streams?

Ans: The exit() function shall then **flush all open streams with unwritten buffered data**, close all open streams, and remove all files created by tmpfile().

1. What happens when you press Ctrl+C?

Ans:

1. I press Ctrl-C in the keyboard.
2. X sends the keyboard event to xterm.
3. xterm translates the Ctrl-C keyboard event and sends it to the pseudo-tty master file descriptor it is holding? (Some magic happens)
4. The kernel detects that some special SIGINT event happens on the pseudo-tty, and finds the session whose controlling terminal is this tty. It sends SIGINT to the current foreground process group of that session, which includes only the processes in our pipeline.
5. What happens when you press Ctrl+Z?

Ans: ctrl z is used to **pause the process**. It will not terminate your program, it will keep your program in background. You can restart your program from that point where you used ctrl z. You can restart your program using the command fg.

1. What is the use of an fd? How is it different from FILE \*?

Ans: A file descriptor (FD, less frequently fildes) is **a unique identifier (handle) for a file or other input/output resource, such as a pipe or network socket**.

Difference between FD and FILE\*

FD: It is an integer value like 0, 1, 2.

The file descriptor is not buffered.

It is less portable and efficient in comparison with the file pointer.

FILE\*: It is a pointer.

The file pointer is buffered.

It is highly portable and efficient.

1. How many fd’s are created for every process? What are they?

Ans: Linux systems limit the number of file descriptors that any one process may open to **1024 per process**.

a file descriptor (FD, less frequently fildes) is **a unique identifier (handle) for a file or other input/output resource, such as a pipe or network socket**.

1. Name the call to get an fd for a file

Ans: Get the file descriptor from a FILE pointer (e.g. file ) in C on Linux: **int fd = fileno(file);** More details can be found in the man page of fileno : fileno manual.

1. If a process creates a child sub process, how can it detect exit of a child?

Ans: You can get the exit status of the child **via the first argument of wait() , or the second argument of waitpid() , and then using the macros WIFEXITED and WEXITSTATUS with it**. waitpid() will block until the process with the supplied process ID exits.

1. Which process reaps the exit code of orphan child?

Ans: The act of reading that exit code is called "**reaping**" the child. Between the time a child exits and is reaped, it is called a zombie.

1. What all does a child inherit from its parent?

Ans: The parent process uses fork() to fork the interpreter. The child process, when it begins, is effectively identical to the parent process. All resources of the parent are inherited by the child process.