1. Which of the following benefits go to multithreaded programming?
A) responsiveness
B) resource sharing
C) economy
D) scalability
E) all of the above

Answer : E

2. _____ is not considered a challenge when designing applications for multicore systems.
A) Deciding which activities can be run in parallel
B) Ensuring there is a sufficient number of cores
C) Determining if data can be separated so that it is accessed on separate cores
D) Identifying data dependencies between tasks.

Answer: B

3. Which of the following is an asynchronous signal?
A) illegal memory access
B) division by zero
C) terminating a process with specific keystrokes
D) none of the above

Answer: C

4. Which of the following options to deliver signals in multithreaded program should be applied to an asynchronous signal that terminates a process (<control><C>, for example)?
A) deliver the signal to the thread to which the signal applies
B) deliver the signal to every thread in the process
C) deliver the signal to certain threads in the process
D) assign a specific thread to receive all signals for the process
E) all of the above

Answer: B

5. Which are included in the context of a thread?

A) register set

B) stacks

C) private storage area

D) all of the above

Answer: D

6. The ready queue can be implemented as a________________.

A) FIFO queue

B) priority queue

C) tree

D) unordered linked list

E) all of the above

Answer: E

7. Which of the following items does not belong to the function of a dispatcher?

A) switching context from one process to another

B) selecting a process among the available ones in the ready queue

C) switching to user mode

D) jumping to the proper location in the user program to resume that program

Answer : B

8. Assume process P0 and P1 are the process before and after a context switch, and PCB0 and PCB1 are respectively their process control block. Which of the following time units are included inside the dispatch latency?

A) P_0 executing

B) save state into PCB_0, and restore state from PCB_1

C) P_1 executing

D) all of the above

Answer: B

9. Which of the following scheduling algorithms must be nonpreemptive?

A) SJF

B) RR

C) FCFS

D) priority algorithms

Answer : C

10. The_______occurs in first-come-first-served scheduling when a process with a long CPU burst occupies the CPU.

A) dispatch latency

B) waiting time

C) convoy effect

D) system-contention scope

Answer: C

11. Which of the following scheduling algorithms gives the minimum average waiting time for a given set of processes?

A) SJF

B) FCFS

C) RR

D) Multilevel queue

Answer : A

12. Why should a web server not run as a single-threaded process?

A web server may be responsible for managing various components:

on the front end: audio, video, text, animation components , etc.
on the back end: scripts to communicate with database, encryption scripts, etc.

Also, a web server needs to respond to its clients calls.

Thus, multi-threading is necessary because we want the components to be able to load in parallel and also respond to multiple client calls at once. If it run as a single-threaded process, then only one client will be served at a given time, resulting in enormous waiting time for a busy server.

13. Multicore systems present certain challenges for multithreaded programming. Briefly describe these challenges.

1. Identifying tasks. This involves examining applications to find areas that can be divided into separate, concurrent tasks. Ideally, tasks are independent of one another and thus can run in parallel on individual cores.

2. Balance: Ensuring that the tasks perform equal work of equal value. For instance, a certain task may not contribute as much value to the overall process as other tasks. Therefore, it's not worth it to use a separate core for that task.
3. Data splitting. Just as applications are divided into separate tasks, the data accessed and manipulated by the tasks must be divided to run on separate cores.

4.Data Dependency: The data accessed by the tasks must be examined for dependencies between two or more tasks.

5.Testing and debugging: and debugging concurrent programs is inherently more difficult than testing and debugging single-threaded applications.

14. Describe two techniques for creating Thread objects in Java.

Java threads can be created by:

- extending thread class

We create a class that extends the java.lang.Thread class. This class overrides the run() method available in the Thread class. A thread begins its life inside run() method. We create an object of our new class and call start() method to start the execution of a thread. Start() invokes the run() method on the Thread object.

- implementing the runnable interface

We create a new class which implements java.lang.Runnable interface and override run() method. Then we instantiate a Thread object and call start() method on this object.

15. In Java, what two things does calling the start() method for a new Thread object accomplish?

Upon the start() method is called, it first allocates memory and initiates a new thread in JVM. Next, it calls the run method, making the thread object eligible to be run be JVM

16. What is a thread pool and why is it used?

The general idea of thread pool is to create a number of threads at start-up and place them into a pool, where they sit and wait for work.

The thread pool is used to solve two problems: 1. the overhead to create a new thread upon every new request 2. the possible issue to exhaust CPU memory without a bound for the concurrently active threads. The thread pool handles the problem by creating a number of threads in a pool first, and when a request occurs, depending on the status of the thread (active/occupied), the thread pool can therefore allocate the resource to the request or put the request in waiting queue, allowing asynchronously request handling.

17. Describe the difference between the fork() and clone() Linux system calls.

The major differences are:

- for clone(),it allows the child process to share parts of its execution context with the calling process, such as the memory space, the table of file descriptors, and the table of signal handlers, while the fork() call does not allow the sharing of *data structures, instead it allocates new memory space for the child process and give it a different memory address*

- clone() syscall executes the function application fn(arg) (This differs from fork(), where execution continues in the child from the point of the original fork() call.) The fn argument is a pointer to a function that is called by the child process at the beginning of its execution. The arg argument is passed to the fn function.

- while fork () is used to create new processes, clone() can also create therads with given parameters, overall, clone() is the base of all fork() syscalls.

18. What role does the dispatcher play in CPU scheduling?

The dispatcher gives control of the CPU to the process selected by the short- term schedular. To perform the task, a context switch between tasks, switch to user mode, and a jump to proper location in the user program is required. Therefore the dispatcher is made to be as fast as possible to reduce the overhead (dispatcher latency) of performing these tasks

19. Explain the difference between response time and turnaround time. These times are both used to measure the effectiveness of scheduling schemes.

Turnaround time is the amount of time elapsed from the time of submission to the time of completion (which includes getting into memory, waiting in the ready queue, CPU execution and I/Os) whereas response time is the average time elapsed from submission until the first response is produced.

20. Explain the process of starvation and how aging can be used to prevent it. Explain the process of starvation and how aging can be used to prevent it.

Starvation occurs when a process is ready to run but is stuck waiting indefinitely for the CPU. This can be caused, for instance, when higher-priority processes which takes a lot of time to execute prevent low-priority processes from ever getting the CPU. Aging involves gradually increasing the priority of a process so that a process will eventually achieve a high enough priority to execute if it has waited for a long enough period of time.