Assignment
MTCS-205: CS-2
Advanced Programming in Unix Environment

By: Chandapu Shiva Krishna
Regd no: 22555
I MTech CS

1. What are the different conditions in which a signal is generated in UNIX? (5M)
Ans: In UNIX, signals are generated in a variety of conditions.

- User-generated signals: Users can send signals to processes using the kill command or by pressing certain key combinations (such as CTRL+C) to interrupt a running process.
- Hardware exceptions: Hardware exceptions such as a segmentation fault or floating-point exception can cause the operating system to send a signal to the offending process.
- Timer signals: The operating system can send timer signals to a process to wake it up after a certain period of time has elapsed.
- Child process events: When a child process terminates or stops, the operating system sends a signal to the parent process.
- System calls: Certain system calls such as wait() can cause a process to block and wait for a signal to be delivered.

2. How does the UNIX kernel handle a signal? (3M)
Ans: The UNIX kernel handles a signal by interrupting a process's normal execution, determining how to handle the signal based on the process's signal disposition (either default or custom), and resuming the process's execution once the signal has been handled. The kernel prioritizes signals based on their signal number and certain signals cannot be ignored or overridden.

3. Name the signals that cannot be ignored by the UNIX kernel, and explain why (3M)
Ans: The two signals in UNIX that cannot be ignored by the kernel are SIGKILL and SIGSTOP. They are designed to be powerful and immediate ways to stop or suspend a process and cannot be caught or overridden because doing so could lead to system instability or insecurity.

4. Write a short note on each of the following functions: (5*3=15M)

A. **kill:** This function sends a signal to a process or group of processes. The signal can be specified by its signal number, and the process or group of processes can be specified by their process ID or process group ID. This function can be used to interrupt a running process, terminate a process, or send other signals for specific purposes.
B. **raise:** This function sends a signal to the calling process itself. The signal can be specified by its signal number, and the calling process can use this function to handle

signals synchronously. For example, a process can use this function to raise a SIGTERM signal to itself to initiate a graceful shutdown.

C. **alarm:** This function sets an alarm to go off after a specified number of seconds. When the alarm goes off, the operating system sends a SIGALRM signal to the calling process. This function is often used to implement time-based operations in UNIX programs.

D. **pause:** This function suspends the calling process until a signal is delivered to it. When a signal is delivered, the process is woken up and the signal is handled. This function is often used in conjunction with signal handlers to implement synchronization between processes.

E. **abort:** This function is used to terminate a process abnormally. When a process calls this function, it causes the process to terminate immediately, without executing any further code. This function is typically used in response to a critical error or unrecoverable condition that cannot be handled in any other way.

5. Write a short note on each of the following signals: (3*3=9M)

A. **Sigpending:** This signal is not a real signal, but rather a function that is used to check whether a process has any pending signals that have not yet been handled. The 'sigpending' function is typically used in conjunction with signal handlers to ensure that all pending signals are handled properly.

B. **Sigaction:** This signal is used to set or retrieve the signal action for a particular signal. The 'sigaction' function is used to install a signal handler function to be executed when a specific signal is received. This function can also be used to retrieve the current signal action for a particular signal.

C. **Sigchld:** This signal is sent to a parent process when one of its child processes terminates. The parent process can then use the 'waitpid' function to retrieve the exit status of the terminated child process. This signal is important for managing child processes in UNIX programs and ensuring that they are properly cleaned up after they terminate.