QUESTION BANK

Introduction

1. Write a C++ program to list the actual values of the following system configuration limits on to a given UNIX OS
   i) Number of clock ticks
   ii) Maximum number of child processes
   iii) Maximum path length
   iv) Maximum number of characters in a file name
   v) Maximum number of open files per process.

2. Write a C++ program to list the actual values of the following system configuration limits on to a given UNIX OS
   i) Number of clock ticks
   ii) Maximum number of child processes
   iii) Maximum path length
   iv) Maximum number of characters in a file name
   v) Maximum number of open files per process.

3. What do you mean by term feature test macros? List all the test macros along with their meaning.

4. Write a C++ program to list the values of the following system configuration.
   i) Maximum number of files which can be opened simultaneously.
   ii) Maximum number of real time signals.
   iii) Maximum value assignable to a semaphore.

5. Write a C++ program to list the actual values of the following system configuration limits on to a given UNIX OS.
   i) Maximum number of files which can be opened simultaneously
   ii) Maximum number of real time signals.
   iii) Maximum value assignable to a semaphore.

6. What is an inode? Why are inode unique only within a file system? How does the OS map an inode to its file name?

7. Write the difference between K&R C and ANSI C.

8. What do you understand by the term feature test macros? List all the five feature test macros along with their meanings.

9. Write a C++ program to list the actual values of the following system configuration limits on to' a given UNIX OS.
   i) Maximum number of child process that can be created
   ii) Minimum number of files that can be opened simultaneously.
   iii) Number of clock ticks.

10. What are the major differences between ANSI "C" and K&R "C"? Explain with examples.

11. What is POSIX standard? Explain the different subsets of POSIX standards.

12. Write a C/C++ POSIX complaint program to check the following limits:
    i) Number of clock ticks
    ii) Maximum number of child processes
    iii) Maximum path length
    iv) Maximum characters in a filename
    v) Maximum number of open files per process

13. Explain the common characteristics of API and describe the error status code.

14. What are the major differences between ANZI C and K&R C? Explain with examples.

15. What is POSIX API? Explain the commonly occurring error status codes and their meaning.

16. Write a C++ program to check and display the POSIX version constant f the system on which it is run.

UNIX Files

1. Describe the UNIX Kernel support for files.

2. Explain the different file types available in UNIX or POSIX system.

3. What is an API? How it is different from C library functions? Why calling an API in more time consuming than calling on user function?

4. List all the file attributes along with their meaning. Which of these attributes can't be changed and why?

5. Explain the different file types available in UNIX or POSIX system.

6. Bring out the differences between hard link and symbolic link.

7. Discuss with a neat diagram the different data structures supported by UNIX Kernel for file manipulation.

8. Discuss the various file types in UNIX or POSIX system.

9. What are the API common characteristics? List any five values of global variables errno along with their meaning whenever API fails.

10. List the difference between hard link and symbolic link.

11. Explain the UNIX Kernel support for files, with a neat diagram.

12. List the commands needed to change the following file attributes:
    i) File size
    ii) User ID
    iii) Last access and modification time
    iv) Hard link count.

13. Discuss the various file types in UNIX or POSIX system.

14. What are the API common characteristics? List any five values of global variables erno along with their meaning whenever API fails.

15. List the difference between hard link and symbolic link. Explain the UNIX Kernel support for files, with a neat diagram.

16. List and explain the different file types available in UNIX.

17. What are the API common characteristics? List any five values of global variables errno along with their meaning whenever API fails.

18. Describe the UNIX Kernel support for files.

19. Explain the different file types available in UNIX or POSIX systems.

20. Describe the UNIX kernel support for files.

21. Differentiate between hard links and symbolic links.

22. Discuss with a neat diagram, the different data structures supported by UNI kernel for file manipulation.

23. List all the attributes of UNIX or POSIX files along with their meaning. Which are the attributes that remain unchanged for the entire life of the file and why?

24. Differentiate between hard link and symbolic links.

<u>**UNIX File APIs**</u>

1. Explain how fcntl API is used for file and record locking.

2. Write the code segment in C that records utmost 100 bytes into a variable but from standard input.

3. Explain the following APIs along with their prototype definitions:
   i) open
   ii) write
   iii) fcntl
   iv) fstat

4. What are symbolic link file APIs? Write a C/C++ program to emulate the UNIX ln command.

5. Give the hierarchy structure of the file classes.

6. With the help of prototype, explain the following API's:
   i) creat
   ii) lseek
   iii) access
   iv) link

7. What is the importance of locking files? What are the mandatory and advisory locks? Why is advisory lock considered safe? What are the draw-backs of advisory lock? Explain.

8. List the structures used to quarry the file attribute in UNIX.

9. Write C++ program to list the following file attributes of given regular file passed as command line argument.
   i) File type
   ii) user ID
   iii) file name
   iv) file size

10. With the help of prototype, explain the following API's:
    i) creat
    ii) lseek
    iii) access
    iv) link

11. List the structures used to quarry the file attribute in UNIX. Write C++ program to list the following file attributes of given regular file passed as command line argument.
    i) File type
    ii) user ID
    iii) file name
    iv) File size

12. What is the importance of locking files? What are the mandatory and advisory locks?

13. Why is advisory lock considered safe? What are the draw-backs of advisory lock? Explain.

14. Explain the following APIs along with their prototype definitions:
    i) open
    ii) write
    iii) fcntl
    iv) fstat

15. Write a C++ program to implement following UNIX commands:
    i) ln
    ii) mv

16. Bring out the differences between hardlink and symbolic link.

17. Explain the importance of file and record locking in UNIX. Show how: fcntl" API can be used for file and record locking.

18. Write a C/C++ program to emulate ln command in UNIX.

19. Write a C/C++ program to emulate mv command in UNIX.

20. Explain the following API's along with their prototype definition and possible cause for failure:

   i) open
   ii) write
   iii) fcntl
   iv) stat

21. How do you access and modify the time stamps of a file? Explain the prototype used for that. Write a program to illustrate the usage of the aboe prototype.

UNIX Processes

1. With a neat diagram, explain the memory layout of C program.
2. What do you mean by command line argument? Explain with an example.
3. Explain the following, with an example:
   i) setjmp and longjmp
   ii) setrlimit and getrlimit
4. What are the different ways in which a process can terminate? Explain with a neat diagram.
5. Explain the memory layout of a C program
6. What are the APIs to query and change the resource limits? List the rules that govern the changing of the resource limits.
7. Explain with a neat block diagram UNIX process data structure.
8. Explain different process termination functions.
9. Explain the setjmp() and longjmp() functions with its prototypes. Illustrate their uses using examples.
10. Explain with a neat diagram, the memory layout of C program.
11. Explain the setjmp() and longjmp() functions with an example C/C++ program illustrating their usage.
12. Explain the use of setjmp() and longjmp() functions, with examples.
13. With related data structure, explain the UNIX kernel support for a process.
14. What are the different ways in which a process can terminate normally?

Process Control

1. What is a job control? What are the three forms of support from the OS required for job control?
2. Explain the special feature of fork API, with suitable example.
3. What is a session? How do you create a session using appropriate shell command?
4. Explain the six different forms of exec API.

5. Explain the fork and vfork functions with example programs.
6. List and explain the family of exec functions with their prototypes. How do they differ from each other? Also give an example program using any one of the exec functions.
7. What is a RACE condition, demonstrate with the program?
8. What is job control? With a neat diagram, explain the job control features.
9. Explain the fork function call with the example programs.
10. What is an orphaned process? Explain with an example.
11. What do you mean by fork() and vfork() functions? Explain both functions with example programs.
12. What is job control? Summarize the job control features with the help of neat diagrams.
13. List and explain the different forms of exec function with prototype declaration along with their meaning. Write a program to echo all its command line arguments and environment variables.
14. What is process accounting? Write a program to illustrate the generation of accounting data.

Signals and Daemon Processes

1. What is the signal mask? Explain with prototype and example.
2. With a neat diagram, explain the method of error logging.
3. What are daemon processes? List their characteristics. Write the rules to code a daemon.
4. Explain the signal handling through the macros along with their meaning and default actions.
5. What is Daemon? Discuss the basic coding rules.
6. Write the demonstration program to show the functions of sigsetjmp and siglongjmp APIs.
7. Explain kill and alarm command with an example program.
8. Give the detailed description and demonstration program of interval timers.
9. What is Daemon? Discuss the basic coding rules.
10. What is Daemon? Describe the daemon process characteristics.
11. Explain the sigaction() function by giving the prototype and discuss its features.

12. Briefly explain the kill() API and the alarm() API.

13. What is a daemon process? Discuss its characteristics.

14. What are signals? List any four signals with brief explanation. Write a program to setup signals handler for SIGALARM and SIGINT signals.

15. What is a daemon processes? Explain the BSD facility adopted b daemon processes for error handling.

16. Write a C++ program to illustrate the implementation of the UNIX Kill command using the Kill API.

Interprocess Communication – 1

1. What do you mean by pipes? List out their limitations. Write a C program that sends "Hello World" message to a child process through the pipes.

2. What is the purpose of message queuing? List and explain message queuing with prototype.

3. What are the three different ways in which client and server process can get access to same IPC structure? Explain with different prototypes.

4. Explain with an example client-server communication using FIFO's.

5. What are the advantages and disadvantages of XSI IPC?

6. What are PIPES? List the 2 limitations of PIPES. Explain how to create a pipe. Write a program to send data from parent to child over a pipe.

7. Explain how client and server will communicate using FIFO's.

8. Explain the following functions related to message queues:
   i) msgget
   ii) msgsnd

9. What is FIFO? Explain how it is used in IPC. Discuss with an example C/C++ program the client-server communication using FIFO's.

10. Write short notes on the following:
    i) Message queues
    ii) Semaphores

11. What are PIPES? Explain the different ways t view a half-duplex pipe. Write a program to create a pipe between a parent and its child and to send data down the pipe.

12. Discuss with an example, the client-server communication using FIFO's.

13. List along with prototype declaration and meaning, the different types of system calls available to create and manipulate semaphores.

Interprocess Communication – 2

1. What is a socket? Describe the socket options. Explain with suitable functions.

2. Write short notes on the following:
   i) Race conditions
   ii) POSIX.l FIPS standard
   iii) Device file API's
   iv) Semaphores.

3. What is byte ordering? Explain the two types of ordering. Explain the APIs to convert between the processor byte order and the network byte for TCP/IP applications.

4. Explain the following APIs with prototypes listen() and accept().

5. Explain the following socket programming functions with their prototypes:
   i) socket
   ii) connect
   iii) listen
   iv) accept

6. Explain the different functions which will be used for exchanging data on sockets.

7. Explain the concept of shared memory with an example C/C++ program.

8. What do you mean by passing file descriptors between processes? Explain.

9. What is a socket? Describe the socket API. Explain the different API's used for establishing connection between two systems using sockets.

10. Write short notes on the following:
    i) Race condition
    ii) File and Record locking.