Q#1, Briefly describe types of System Calls?

Answer:

1. **Process control**
    1. End, abort
    2. Load, execute
    3. Create, data
    4. Get attribute, set attribute
    5. Wait time
    6. Wait event
    7. Allocate free space

2. **File management**

    1. Create, delete
    2. Open, close
    3. Load, execute
    4. Get file attribute, set file attribute

3. **Device management**

    1. Requst, release
    2. Read, write
    3. Get attribute, set attribute
    4. Attach, deattach

4. **Information maintance**

    1. Set data, set time

    2. Set system state, get system state

    3. Get process, file, device id, set process, file, device id

5. **Communication**

    There are two common model of the interprocess communication: the massage-passing model and the shared-memory model. In **massage-passing model** the communicating process exchange the message with one another to transfer information. In the **shared-memory model** processes use shared memory create and shared memory attach system call to create and gain access to regions of memory owned by other processes.
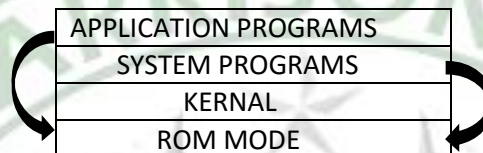
6. **Protection**

    1. Set permission, get permission

    2. Allow user, deny user.

Q#2, Do MS-DOS OS and Unix OS use different Architecture?

Answer:

MS-DOS

- Interfaces and functionality are not separated
- Application programs are able to access kernel

APPLICATION PROGRAMS
SYSTEM PROGRAMS
KERNAL
ROM MODE

UNIX

- Unix has two part of kernel (it is separation of kernel) with is kernel and system program
- Kernel is further separated into series of interface and device drivers

USER
SHELL
SIGNAL FILE **CPU** HANDLING SHCHUDLE
KERNEL CALL TO HEADER
DEVICE CONTROL | MEMOREY CONTROL | TERMINAL CONTROL

Q#3, What is the difference between Mechanism and Policy in OS ?

Answer:

Mechanism is define what the os should do. All the working of os. Policy describe how treat work or task to be done..

Q#4, Do Monolithic architecture provides more security to kernel?

Answer:

NO, because kernel is accessible and can be modified. All the functionality of the kernel is performed on the same level.

Q#5, Why user is at the last layer in layered architecture?

Answer:

In layered architecture user is at last layer because in layered base architecture operating system user is the first priority of os. User don't know about the working of the os witch have different operation like i/o operation kernel functionality. User just putt the command or do the instruction and receive the result in the architecture user did know about the process that instruction which he passed how to execute how many process are include, how many i/o devices are include tu execute his instruction. In other words we can say user in layered architecture operating system has no interact directly with hardware..

The end
☺