Answer to the question no: 1

Orphan Process:
When the parent process terminates before the child process, the child process becomes the orphan process. Then the child process becomes the sub process of os. Usually when wait() is not used in the parent process, this incident occurs.

Zombie Process:
When a child process terminates before the wait() is executed in the parent process, this state happens. At this time the child process is in a zombie state and called the zombie process.

Answer to the question no: 2

Sub-thread:
A thread which is subordinate to another thread is known as a sub-thread. That means a thread which is executed under another thread is known as a sub-thread. In OS, thread is a segment of a process that is executed. It is used for multi-tasking. A thread is executed by the core of a processor. Thread uses shared memory. A thread is created using the pthread_create() statement.

Child process:
Child process is created using the fork() statement. When fork() is executed, a parent process and child process are created in a program. A process can have multiple threads. Process uses isolated memory.

Answer to the question no: 3

Process ID:
Process ID or PID means process identifier. Every process has its own identifier and through this identifier a process can be easily identified. This ID is unique and is generated by this OS.

Thread ID:
Similarly every thread has its own unique number which is known as thread ID and through using it a thread can be easily identified. This ID is also generated by the OS.

Answer to the question no: 4

Unnamed Pipe:
Pipe is a way of communication between two processes. Ordinary pipe or “unnamed pipe” lasts as long as the process is alive or continues to be executed. When the program terminates or stops communicating, it no longer exists. It is created using pipe() statement.

Named Pipe:
Named pipe is more powerful. It does not terminate after the process is finished or stops communicating. So, other processes can use it. It lasts as long as the system is alive or it is not deleted. Once it is created, it appears in the file system. It is created using mkfifo() statement.

Answer to the question no: 5

User Process:
User process is the process for the user program. It can access its own instruction and data but it can not access the system / kernel / machine level instruction and data. For example- when a user writes a program, it creates a user process. User initiates the user process.

System Process
System processes can access both instruction and data of the user process and the system / kernel. For example - OS uses system processes. OS initiates the system process.

Answer to the question no: 6

Shared Resource
Shared resources are those resources which can be accessed by all who have the permission of accessing them. For example- if multiple threads are created in a single process, all the threads can have access to the global instances of that process. These global instances are here shared resources.

Non-Shared Resource:
Non-shared resources are those that can not be accessed by others rather than the owner. In previous example of multiple threads in a single process, all the threads can have their local instances and the process will allocate different memory locations for each of the threads. Here each individual thread can not access the other thread's information. So this can be called non-shared resources.

Answer to the question no: 7

Preemptive Scheduling:
Here the time slot of the CPU is divided into processes. In this time an individual process might complete the execution or might not be able to do it. When a process switches from running state to ready state or from waiting state to ready state, preemptive scheduling occurs. If the process is not finished execution, it remains in the ready queue. If a process is in the ready queue, it gets its chance to be executed next. Algorithms used in this scheduling are - Round Robin (RR),Shortest Remaining Time First (SRTF), Priority (preemptive version), etc.

Non-Preemptive Scheduling:
In this technique, a process gets enough time until it gets terminated or it goes to the waiting state. Here no process gets any interrupts until it is completed and then the processor switches

to another process. Algorithms used in this scheduling are: Shortest Job First (SJF basically non preemptive) and Priority (non preemptive version), etc.