# Conceptual Question's Answers

1. Signals are one of the most simplified form of **IPC** mechanism, where no data transfer occurs. Signals are asynchronous interrupt of the target process or thread which will have to handle the signal with a specific routine during non-atomic operations. Signals are started by the kernel and handled by the processes. The computational and memory footprint of signals are hence small.

2. `SIGINT` - This is the signal sent when we want to interrupt a process, from the terminal you can use **Ctrl-C**, to send the signal to the current process.

   `SIGTSTP` - This is the signal sent to stop a process for now in manner so it can be resumed later. We can use **Ctrl-z** to send a `SIGTSTP` to the current process. We can also use the **kill** utility to send this signal like this:

   ```
   kill -SIGSTOP [pid]
   ```

   `SIGCONT` - This is the signal sent to continue or restart a process that was stopped using `SIGSTP`. We can use the **kill** utility to send this signal like this:

   ```
   kill -SIGCONT [pid]
   ```

3. `kill()` - is a system call used to send any signal to a process or processes the current program have permission for. In **C** a signal can be sent using the following syntax. `kill(pid,sig)`, where `pid` is the process id and `sig` is the signal we are trying to sent.

   `waitpid()` - is a system call used to wait for a child process to change its state. The state change can be a termination or being stopped or being resumed after stopping. When using `waitpid(pid,status,options)`, `pid` is the process id the child-process being waited for, `status` is the type of exit status or state changes we are waiting for and `options` specify optional actions for the waiting which can also be a combination of many options.

   To terminate a child process and wait until it has ended we can do something like this in **C**:

```
//Create a child
int status;
pid_t child2;
child1 = fork();
// Do something with child
...
// Child is dead
waitpid(-1, &status, 0); // Wait until child is dead here
```