# What is a process?

A process is a program that is currently in execution. In multi programming, a batch system runs jobs, while a time-shared multitasking OS will run processes or tasks. A program will become a process once it has been loaded into memory. When it has been loaded into memory, it will join other processes, that depending on the architecture style of the OS, will access the CPU in order to properly execute. One program however, can have multiple processes.

# How is a process organized in memory?

A process is stored with a text section, a data section, a stack section and a heap section. The Text section stores the actually program code, while the data section will story global variables, the heap stores dynamically allocated memory chunks, and the stack holds function parameters, return variables, and return addresses.

# Process states.

There are 5 states for the process to be in. The first is the new state. This tells the OS that this is a new process that is waiting to be placed in the ready state. The ready state is a state in which the OS gives the CPU to this process. After the CPU has been assigned to a process, the process now enters the running state, which indicates that the process is currently executing. If the process requests an I/O signal, the process is placed into the waiting state. The process can also enter the waiting state if the process is waiting on a signal from another external source. If the process finishes, or causes an exception, the process will be place in a terminated state. This state indicates to the OS that this process is no longer running.

# The Process Control Block

# 