**Task 1**

* The process will be running on a general-purpose system.
* Processes are the context associated with a program in execution.
* A process has a life cycle. Once it’s finished its execution, it terminates the entire process.
* A process can switch between different stages, such as running, ready, blocked, etc.
* The OS stores a priority queue of process contexts that are ready for execution.
  + I’ll use a range of 0 to 32767 for the process ID.
  + Process Allocation Python Code:



* + When the program runs out of PIDs, it simply returns -1.
* The context stores information about the process such as:

Process Context Table

|  |  |
| --- | --- |
| **Name** | **Data Structure Used** |
| Stack | stack |
| Heap | heap |
| Code Area | string (data pointer) |
| Data Area | string (data pointer) |
| Register States | array (each index corresponds to a register) |
| Process ID | int |
| Parent Process ID | int / None |
| User ID | int |
| Current Directory | string |
| SP | string (data pointer) |
| PC | string (data pointer) |
| Priority | int |
| State | string (ready/blocked/running) |

**Task 2**

State changes

|  |  |  |  |
| --- | --- | --- | --- |
| **Current State** | **Event** | **New State** | **Comment** |
| Ready | Reaches top of queue | Running | The process receives it’s allocated time slot for the CPU |
| Running | I/O operation that must be awaited | Blocked |  |
| Blocked | I/O operation finishes | Ready |  |
| Running | The process runs out of time on the CPU | Ready |  |

* Ready refers to the process being ready to be run, however it is waiting to get its timeshare of the CPU.
* Running means, it’s currently running on the CPU.
* Blocked means the process has been put on hold until an event completes (generally an I/O operation)

­­­­­­­