`

1. **Objects and their attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| Object | Class name | Destription | Attributes |
| CPU Scheduler | CpuScheduler | Class containing all other objects implementing every attribute of simulation | Int turnaroundTime  Int cpuUtilization  Int throughput  Int noEventTriggerFlag  Void eventScanner |
| Process | Process | Class generating processes with arbitrary PGT, CET, IOT, IOD for certain range | Int procGenTime  Int cpuExecTime  Int CallTimeIO  Int deviceIO  int processID  static int processID  int getProcGenTime  int getCpuExecTime  Int getCallTimeIO  Int getDeviceIO  Int getProcessID |
| ‘Ready’ queue | ReadyQueue | Queue of all processes waiting for CPU | Int queueSize  Int getQueueSize |
| Block of CPUs | CpuBlock |  | Void processing()  Int processorNumber  Int getProcessorNumber |
| I/O device queues | DeviceQueue | Vector of vectors containing queues of each I/O Device | Int queueSizeIO |
| I/O Devices | DeviceIO | Each I/O device | int deviceID\* |
| Analyser | Analyser | Gathers data and analyses it In form of histogram |  |
| Generator | Generator | Implement uniform and xxXXXxxx distribution |  |

1. **Description of time and conditional events:**

Time events:  
- generation of process

- termination of process on CPU

- termination of process on I/O device

Conditional events:

- starting work on processor

- starting work on I/O Device

|  |  |
| --- | --- |
| Event | Algorithm |
| Generation of process | 1. Generate process with assigned values of PGT, CET, IOT, IOD and unique ID 2. Move it to tail of the **ready** queue |
| Termination of process on CPU | 1. Check if conditions for termination of process are fulfilled 2. Make CPU free 3. Pass process to I/O queue |
| Termination of process on I/O device | 1. Check if conditions for termination of process are fulfilled 2. Make I/O Device free 3. Pass process to **ready** queue |

|  |  |
| --- | --- |
| Event | Algorithm |
| starting work on CPU | 1. Move process from ‘ready’ queue to free CPU 2. Assign CPU to be in ‘busy state’ 3. Perform operation on a process |
| starting work on I/O Device | 1. Move process from I/O Device queue to ‘ready’ queue 2. Assign I/O to be in ‘busy state’ 3. Perform operation on I/O Device |

**3. Description of processes and block diagram**

1. Process generation:
2. Create a process
3. Enter process to **ready queue**
4. Schedule next process generation
5. Start working on CPU:
6. Check if CPU is free
7. Assign process to free CPU
8. Leave **ready queue**
9. Assign CPU to be **busy** and process to be in **running** state, start execution on process
10. Termination of process on CPU device:
11. Check if call to I/O device occurred.   
    If yes:

* Change process state from **running** to **waiting**
* Move process with **waiting** state to device queue

If no:

* Execute the process’s instructions
* Remove process from the OS
* Set the CPU flag to ‘waiting for next process’

1. Starting work on I/O Device:
2. Check if required I/O Device is idle

If yes:

- Assign process from its **device queue** to previously mentioned device

- Perform operation on I/O

If no:

* Stay in **device queue** and do nothing

1. Termination of process on I/O:
2. After completion of I/O, switch process state from **waiting** to **ready**
3. Move process with **ready** state to **ready queue**
4. Set I/O to idle

