**2 - Test Cases**

**Team:**

AJ Omartian - Project Manager

Danya Alrawi and Ikbel Amri - Architect

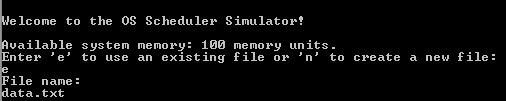
Elisabeth Fernandez - Business Analyst

Alex Hernandez - Lead Software Developer

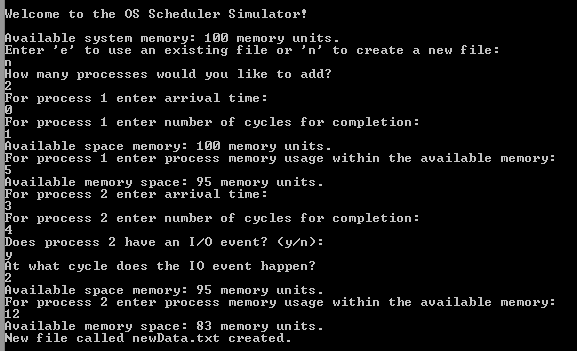
**Reading Process List:**

The program reads a list of process definitions inputted from either:

* An existing file



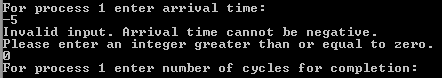
* A list of processes inputted in the user interface



We can see that the user is not asked to specify an IO event for the first process because it takes one cycle to complete. This was a design decision we agreed on, that processes with one cycle cannot have an IO event.

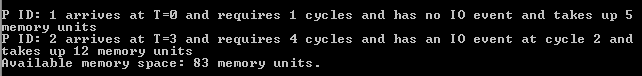
**Verifying Data:**

Verifying the validity of input was a crucial requirement for the functioning of the simulator. For that purpose, we prompted the user to re enter the input values if they did not meet the required specifications. Example:

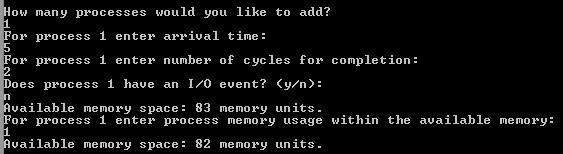


**Displaying, Adding, Deleting and Making Updates to Process List:**

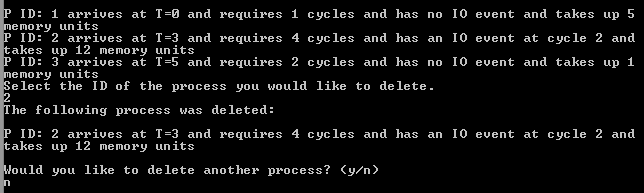
* Displaying the list of active processes: The user has the ability to display the list of active processes by pressing (1) from the main menu after inputting the list:



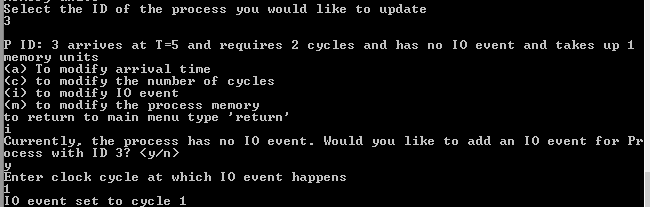
* Adding processes: The user has the ability to add additional processes to the list of active processes by pressing (2) from the main menu:



* Deleting processes: The user has the ability to delete processes from the list of active processes by pressing (4) from the main menu. They will be shown a the list of active processes, asked to enter the ID of the process to delete, then asked if they would like to delete other processes:



* Updating the active list of processes: The user can update any process from the active list of processes by pressing (3) through a thorough menu. Example of updating the IO event:



**First Come First Serve Algorithm and Memory Management:**

* FCFS is a non-preemptive algorithm, therefore processes do not exit until they are complete (i.e. until they run out of clock cycles)



**Round Robin Algorithm and Memory Management:**

* This algorithm is pre-emptive, a process it taken out once the time quantum is reached, or the process has an IO event, in which case it is taken out and the next process becomes active.
* Time Quantum: The time quantum value was defined as part of the setup of the environment.
* Dispatch Penalty: The user has the ability to set the Dispatch penalty which is the time penalty for the Dispatcher to update the running process.

