



Faculty of Engineering - Cairo University

Credit Hours System

Fall 2022



Cairo University

CMPN303: Operating System Project - Phase 1

Submitted to:

- Eng Mohamed Abd El Kareem
- Eng Samir Hosny

Data Structures	3
PCB class:	3
Linked Queue:	3
Linked Priority Queue	3
Multilevel Queue	4
Algorithms Explanation and Results	5
SJF	5
Explanation	5
Result	5
Pre-emptive HPF	6
Explanation	6
Result	6
RR	7
Explanation	7
Result	7
Multi-level Feedback Loop	8
Explanation	8
Result	9
Data Structures	10
Memory	10
Modifications from Last Discussion's feedback	11
Scheduling Techniques Modifications	11
Results	13
Workload Distribution	17

Phase 1

Data Structures

PCB class:

This is where we keep the process data

```
{  
  Id , remaining_time , current_state , finish_time , start_time , arrival_time , execution time  
}
```

- ***current_state is an enum : (Running , Blocked , Finished)***

Linked Queue:

We used queue as it is the most appropriate data structure for the first come first served technique needed in the multi-level feedback loop scheduling technique as well as using it as a circular queue in the round-robin scheduling technique (*Which will be explained in the next few pages*)

The class includes the following data members :

- **Node Struct that holds the process data and pointer to next node**
- **Member functions : enqueue , dequeue , isEmpty , destroyQueue and peek**

Linked Priority Queue

The purpose of the priority queue was to safely implement both pre-emptive high priority first (HPF) scheduling technique , the shortest job first (SJF) scheduling technique and the constructing buffer for the multi-level feedback loop technique. It has almost the same members of queue ***in addition to the priority variable***

Multilevel Queue

This data structure is simply an array of queues of size 11 [representing priorities from 0 to 10]

It uses the same member functions of the queue by just indexing the queue at priority p

For example : The DS has the function named Enqueue_Multilevel(multilevel *q)

The function calls the “enqueue” function of the queue data structure as follows

Enqueue(q->Qptr[p]);

An extra member function named “Multilevel_isEmpty()” returns an integer data type representing the first filled queue (it's guaranteed it's the highest priority queue)

So the structure is as follows

Struct {

*Queue *Qptr[11];*

}; Multilevel_Queue

Member functions :

Void Enqueue_Multilevel , bool Dequeue_Multilevel , void Peek_multilevel , int isEmpty_Multilevel and void destroy_Multilevel

Algorithms Explanation and Results

SJF

Explanation

The function uses the priority level queue with minimum execution time as high priority to enqueue. Since it's a non-preemptive scheduling algorithm, it's never interrupted until the process inside is finished. So, the context switching happens **only** when the current time of the process ends. It exits on condition that no process in the PCB and the count of process is matching to the current count of processes.

Result

2	At time 3	process 1	started arr	3	total	3	remain	3	wait	0		
3												
4	At time 6	process 1	finished arr	3	total	3	remain	0	wait	0	TA	3 WTA
5	1.00											
6	At time 6	process 5	started arr	6	total	1	remain	1	wait	0		
7												
8	At time 7	process 5	finished arr	6	total	1	remain	0	wait	0	TA	1 WTA
9	1.00											
10	At time 7	process 2	started arr	4	total	5	remain	5	wait	3		
11	At time 12	process 2	finished arr	4	total	5	remain	0	wait	3	TA	8 WTA
12	1.60											
13	At time 12	process 3	started arr	5	total	6	remain	6	wait	7		
14	At time 18	process 3	finished arr	5	total	6	remain	0	wait	7	TA	13 WTA
15	2.17											
16	At time 18	process 8	started arr	17	total	2	remain	2	wait	1		
17	At time 20	process 8	finished arr	17	total	2	remain	0	wait	1	TA	3 WTA
18	1.50											
19	At time 20	process 9	started arr	19	total	5	remain	5	wait	1		
20	At time 25	process 9	finished arr	19	total	5	remain	0	wait	1	TA	6 WTA
21	1.20											
22	At time 25	process 10	started arr	19	total	5	remain	5	wait	6		
23	At time 30	process 10	finished arr	19	total	5	remain	0	wait	6	TA	11 WTA
24	2.20											
25	At time 30	process 4	started arr	6	total	7	remain	7	wait	24		
26	At time 37	process 4	finished arr	6	total	7	remain	0	wait	24	TA	31 WTA
27	4.43											
28	At time 37	process 6	started arr	9	total	8	remain	8	wait	28		
29	At time 45	process 6	finished arr	9	total	8	remain	0	wait	28	TA	36 WTA
30	4.50											
31	At time 45	process 7	started arr	13	total	9	remain	9	wait	32		
32	At time 54	process 7	finished arr	13	total	9	remain	0	wait	32	TA	41 WTA
33	4.56z											
34	CPU utilization = 94.55%											
35	Avg WTA = 2.42											
36	Avg Waiting = 10.2											

Pre-emptive HPF

Explanation

The function uses the priority level queue with least priority number as high priority to enqueue. Since it's a preemptive scheduling algorithm, it's interrupted when a process of higher priority reaches the system in the receive message function, enqueues itself in the queue so the first process is not the current process (the one to be interrupted) so the current is blocked and the new process enters and works and so on. So, the context switching happens **only** when the current process's priority is more than that of the new process. It exits on same condition as the SJF.

Result

```
At time 3    process 1    started arr    3    total 3    remain 3    wait 0
At time 6    process 1    finished arr    3    total 3    remain 0    wait 0    TA    3 WTA 1.00
At time 6    process 5    started arr    6    total 1    remain 1    wait 0
At time 7    process 5    finished arr    6    total 1    remain 0    wait 0    TA    1 WTA 1.00
At time 7    process 4    started arr    6    total 7    remain 7    wait 1
At time 9    process 4    stopped arr    6    total 7    remain 5    wait 0
At time 9    process 6    started arr    9    total 8    remain 8    wait 0
At time 17   process 6    finished arr    9    total 8    remain 0    wait 0    TA    8 WTA 1.00
At time 17   process 4    resumed arr    6    total 7    remain 5    wait 6
At time 22   process 4    finished arr    6    total 7    remain 0    wait 9    TA    16 WTA 2.29
At time 22   process 10   started arr    19   total 5    remain 5    wait 3
At time 27   process 10   finished arr    19   total 5    remain 0    wait 3    TA    8 WTA 1.60
At time 27   process 3    started arr    5    total 6    remain 6    wait 22
At time 33   process 3    finished arr    5    total 6    remain 0    wait 22    TA    28 WTA 4.67
At time 33   process 9    started arr    19   total 5    remain 5    wait 14
At time 38   process 9    finished arr    19   total 5    remain 0    wait 14    TA    19 WTA 3.80
At time 38   process 2    started arr    4    total 5    remain 5    wait 34
At time 43   process 2    finished arr    4    total 5    remain 0    wait 34    TA    39 WTA 7.80
At time 43   process 7    started arr    13   total 9    remain 9    wait 30
At time 52   process 7    finished arr    13   total 9    remain 0    wait 30    TA    39 WTA 4.33
At time 52   process 8    started arr    17   total 2    remain 2    wait 35
At time 54   process 8    finished arr    17   total 2    remain 0    wait 35    TA    37 WTA 18.50
CPU utilization = 94.55%
Avg WTA = 4.60
Avg Waiting = 14.7
[SCHEDULER] Freeing resources...
```

RR

Explanation

The round robin technique is a non preemptive technique (I mean not interrupted by priority) as it's basically depends on the quanta given to the running process and hereby, there are two cases to be handled

Case1 : Process finished before or on it's quantum.

What happens ? The process is dequeued from the whole queue and never enters again and hence , marked as finished **and the quantum is reset to 0**

Case 2: Process didn't finish at its quantum and will have to run again

Then we mark it as stopped , save its status , dequeue it and enqueue it and peek the new process to start **and the quantum is reset to 0 (it's a variable)**

Otherwise , the quantum passed is incremented and remaining time of process is decremented

Result

At time 3	process 1	started arr	3	total	3	remain	3	wait	0		
At time 6	process 1	finished arr	3	total	3	remain	0	wait	0	TA	3 WTA 1.00
At time 6	process 2	started arr	4	total	5	remain	5	wait	2		
At time 10	process 2	stopped arr	4	total	5	remain	1	wait	0		
At time 10	process 3	started arr	5	total	6	remain	6	wait	5		
At time 14	process 3	stopped arr	5	total	6	remain	2	wait	0		
At time 14	process 4	started arr	6	total	7	remain	7	wait	8		
At time 18	process 4	stopped arr	6	total	7	remain	3	wait	0		
At time 18	process 5	started arr	6	total	1	remain	1	wait	12		
At time 19	process 5	finished arr	6	total	1	remain	0	wait	12	TA	13 WTA 13.00
At time 19	process 6	started arr	9	total	8	remain	8	wait	10		
At time 23	process 6	stopped arr	9	total	8	remain	4	wait	0		
At time 23	process 2	resumed arr	4	total	5	remain	1	wait	18		
At time 24	process 2	finished arr	4	total	5	remain	0	wait	15	TA	20 WTA 4.00
At time 24	process 7	started arr	13	total	9	remain	9	wait	11		
At time 28	process 7	stopped arr	13	total	9	remain	5	wait	0		
At time 28	process 3	resumed arr	5	total	6	remain	2	wait	21		
At time 30	process 3	finished arr	5	total	6	remain	0	wait	19	TA	25 WTA 4.17
At time 30	process 8	started arr	17	total	2	remain	2	wait	13		
At time 32	process 8	finished arr	17	total	2	remain	0	wait	13	TA	15 WTA 7.50
At time 32	process 9	started arr	19	total	5	remain	5	wait	13		
At time 36	process 9	stopped arr	19	total	5	remain	1	wait	0		
At time 36	process 10	started arr	19	total	5	remain	5	wait	17		
At time 40	process 10	stopped arr	19	total	5	remain	1	wait	0		
At time 40	process 4	resumed arr	6	total	7	remain	3	wait	31		
At time 43	process 4	finished arr	6	total	7	remain	0	wait	30	TA	37 WTA 5.29
At time 43	process 6	resumed arr	9	total	8	remain	4	wait	30		
At time 47	process 6	finished arr	9	total	8	remain	0	wait	30	TA	38 WTA 4.75
At time 51	process 7	stopped arr	13	total	9	remain	1	wait	0		
At time 51	process 9	resumed arr	19	total	5	remain	1	wait	31		
At time 52	process 9	finished arr	19	total	5	remain	0	wait	28	TA	33 WTA 6.60
At time 52	process 10	resumed arr	19	total	5	remain	1	wait	32		
At time 53	process 10	finished arr	19	total	5	remain	0	wait	29	TA	34 WTA 6.80
At time 53	process 7	resumed arr	13	total	9	remain	1	wait	39		
At time 54	process 7	finished arr	13	total	9	remain	0	wait	32	TA	41 WTA 4.56

Avg WTA = 5.77

Avg Waiting = 20.8

Multi-level Feedback Loop

Explanation

“It sounds easy , easy until you code it”

A for loop is used for $i = \text{current_occupied_level}$ till $i = 10$

We have three cases at the enqueue stage

Case 1 : A new process arrives in an empty multilevel

What happens: it enters as a current process in its specified level and keep running until finishing or until its quantum is finished.

Case 2 : A new process with less priority arrives while working on current process

What happens: the current process continue running until finishing or until end of quantum and the new process is enqueued at its specified level (special case : new_process_level is equal to $\text{current_process_level}$ will be discussed on blocking scenarios)

Case 3: A new process with high priority arrives while working on current process

*What happens: **No Interrupts !** The current process continue running until quantum or until finishing*

Blocking cases:

After blocking , the $\text{current_occupied_level}$ is recalculated and the quantum is set to 0

Case 1: A process is blocked (because it didn't finish its quantum) and can be degraded to another level

Case 1.1 : The level is not empty and the iterator i (representing the current level) is less than or equal the new calculated $\text{current_occupied_level}$

Hence dequeue current process from level i to level $i+1$, and peek the next process in the level and run it

Case 1.2 : The level is not empty and the iterator i (representing the current level) is greater the new calculated $\text{current_occupied_level}$

Hence dequeue current process from level i to level $i+1$, and peek the next process in the new_level and run it (Assumption here : if I returned to a higher level this means the process is started)

Case 1.3: No process except for this process

So, it continues to work on condition that it degrades itself to the lower level

Case 2: A process is blocked at the last level

if $i == 10$, then the process is not enqueued in the multi-level , instead , it's kept in a buffer named “reconstructing buffer” that will refill the multilevel queue with the correct priorities and remaining time

Finishing cases:

Process dequeues itself from the multilevel forever and peeks to next process and quantum is set to 0

Case 1 : Finished without remaining processes

making the same checks it does on blocking state except that it can return -1 for occupied level and no currentProcess running , hence we are done or counting idle cycles if current_process_count is not equal total_process_count

Case 2: Finished without remaining process

Same conditioning on blocking state to check the highest priority level through the variable current_occupied_level and start from there.

Result

At time 3	process 1	started arr	3	total	3	remain	3	wait	0		
At time 6	process 1	finished arr	3	total	3	remain	0	wait	0	TA	3 WTA 1.00
At time 6	process 4	started arr	6	total	7	remain	7	wait	0		
At time 10	process 4	stopped arr	6	total	7	remain	3	wait	0		
At time 10	process 6	started arr	9	total	8	remain	8	wait	1		
At time 14	process 6	stopped arr	9	total	8	remain	4	wait	0		
At time 14	process 5	started arr	6	total	1	remain	1	wait	8		
At time 15	process 5	finished arr	6	total	1	remain	0	wait	8	TA	9 WTA 9.00
At time 15	process 6	resumed arr	9	total	8	remain	4	wait	2		
At time 19	process 6	finished arr	9	total	8	remain	0	wait	2	TA	10 WTA 1.25
At time 19	process 10	started arr	19	total	5	remain	5	wait	0		
At time 23	process 10	stopped arr	19	total	5	remain	1	wait	0		
At time 23	process 3	started arr	5	total	6	remain	6	wait	18		
At time 27	process 3	stopped arr	5	total	6	remain	2	wait	0		
At time 30	process 4	finished arr	6	total	7	remain	0	wait	17	TA	24 WTA 3.43
At time 30	process 10	resumed arr	19	total	5	remain	1	wait	10		
At time 31	process 10	finished arr	19	total	5	remain	0	wait	7	TA	12 WTA 2.40
At time 31	process 3	started arr	5	total	6	remain	2	wait	26		
At time 33	process 3	finished arr	5	total	6	remain	0	wait	22	TA	28 WTA 4.67
At time 33	process 9	started arr	19	total	5	remain	5	wait	14		
At time 37	process 9	stopped arr	19	total	5	remain	1	wait	0		
At time 37	process 2	started arr	4	total	5	remain	5	wait	33		
At time 41	process 2	stopped arr	4	total	5	remain	1	wait	0		
At time 41	process 9	resumed arr	19	total	5	remain	1	wait	21		
At time 42	process 9	finished arr	19	total	5	remain	0	wait	18	TA	23 WTA 4.60
At time 42	process 2	started arr	4	total	5	remain	1	wait	38		
At time 43	process 2	finished arr	4	total	5	remain	0	wait	34	TA	39 WTA 7.80
At time 43	process 7	started arr	13	total	9	remain	9	wait	30		
At time 47	process 7	stopped arr	13	total	9	remain	5	wait	0		
At time 47	process 8	started arr	17	total	2	remain	2	wait	30		
At time 49	process 8	finished arr	17	total	2	remain	0	wait	30	TA	32 WTA 16.00
At time 49	process 7	resumed arr	13	total	9	remain	5	wait	31		
At time 53	process 7	stopped arr	13	total	9	remain	1	wait	0		
At time 53	process 7	resumed arr	13	total	9	remain	1	wait	39		
At time 54	process 7	finished arr	13	total	9	remain	0	wait	32	TA	41 WTA 4.56
CPU utilization = 94.23%											

```
At time 54 process 7 finished arr 13 total 9 remain 0 wait 32 TA 41 WTA 4.56
CPU utilization = 94.23%
Avg WTA = 5.47
Avg Waiting = 17.0
```

Phase 2

Data Structures

Memory Modifications from Last Discussion's feedback

- Fixing clock synchronization error using "SIGSTOP" and "SIGCONT" and making the clock starts from 0
- Fixed the utilization calculation due to clock starting from -1
- Adjusting the code written to meet our assumption in the multi-level feedback loop [On finishing quantum or remaining time, the process arriving in the message queue with highest priority wasn't seen due to synchronization problem]

Scheduling Techniques Modifications

- Added the allocation and deallocation part to each scheduling algorithm.**How?**
Processes are allocated on arrival if their request from memory can be processed (i.e: there's a place in the memory). Failed allocation results in putting the process in a buffer (customized buffer) until the deallocation of another process occurs in memory. Checking all over this buffer for a process that can be allocated and runned.

Our Customizations to the scheduling techniques are as follow:

- 1) SJF : the buffer is priority buffer according to the execution time , yet it still iterates on the whole buffer for possible allocations in memory**
- 2) HPF: the buffer is priority buffer to priority , it has a similar assumption to SJF concerning the whole iteration**
- 3) The round-robin has its buffer's priority normalized (all equal) this to assure the concept of FCFS**
- 4) MLFQ technique has the same assumption of HPF buffer**

Results

Test case

```
#id arrival runtime priority memsize
1    1     5     2    321
2    3     1     3    129
3    4     3     1    700
4   10     4     4     31
5   11     2     1    300
```

1) SJF

```
At time 1 allocated 512 bytes for process 1 from 0 to 511
At time 1 process 1 started arr 1 total 5 remain 5 wait 0
At time 3 allocated 256 bytes for process 2 from 512 to 767
[SCHEDULER] Failed to allocate memory for process [3]: Memory requested 700
At time 6 freed 512 bytes from process 1 from 0 to 511
At time 6 process 1 finished arr 1 total 5 remain 0 wait 0 TA 5 WTA 1.00
At time 6 process 2 started arr 3 total 1 remain 1 wait 3
At time 7 freed 256 bytes from process 2 from 512 to 767
At time 7 process 2 finished arr 3 total 1 remain 0 wait 3 TA 4 WTA 4.00
At time 7 allocated 1024 bytes for process 3 from 0 to 1023
At time 7 process 3 started arr 4 total 3 remain 3 wait 3
[SCHEDULER] Failed to allocate memory for process [4]: Memory requested 31
At time 10 freed 1024 bytes from process 3 from 0 to 1023
At time 10 process 3 finished arr 4 total 3 remain 0 wait 3 TA 6 WTA 2.00
At time 10 allocated 32 bytes for process 4 from 0 to 31
At time 10 process 4 started arr 10 total 4 remain 4 wait 0
At time 11 allocated 512 bytes for process 5 from 512 to 1023
At time 14 freed 32 bytes from process 4 from 0 to 31
At time 14 process 4 finished arr 10 total 4 remain 0 wait 0 TA 4 WTA 1.00
At time 14 process 5 started arr 11 total 2 remain 2 wait 3
At time 16 freed 512 bytes from process 5 from 512 to 1023
At time 16 process 5 finished arr 11 total 2 remain 0 wait 3 TA 5 WTA 2.50
[SCHEDULER] Current Time is 17
[SCHEDULER] Stopping SJF...
[SCHEDULER] Terminated normally...
[SCHEDULER] Logging output...
CPU utilization = 88.89%
Avg WTA = 2.10
Avg Waiting = 1.8
[SCHEDULER] Freeing resources...
[SCHEDULER] Freeing resources...Clock Terminating!...[PROCGEN] Terminating...
```

2) HPF

```
[SCHEDULER] Starting HPF...
At time 1 allocated 512 bytes for process 1 from 0 to 511
At time 1      process 1      started arr    1      total    5      remain    5      wait    0
At time 3 allocated 256 bytes for process 2 from 512 to 767
[SCHEDULER] Failed to allocate memory for process [3]: Memory requested 700
At time 6 freed 512 bytes from process 1 from 0 to 511
At time 6      process 1      finished arr    1      total    5      remain    0      wait    0      TA      5 WTA    1.00
At time 6      process 2      started arr    3      total    1      remain    1      wait    3
At time 7 freed 256 bytes from process 2 from 512 to 767
At time 7      process 2      finished arr    3      total    1      remain    0      wait    3      TA      4 WTA    4.00
At time 7 allocated 1024 bytes for process 3 from 0 to 1023
At time 7      process 3      started arr    4      total    3      remain    3      wait    3
[SCHEDULER] Failed to allocate memory for process [4]: Memory requested 31
At time 10 freed 1024 bytes from process 3 from 0 to 1023
At time 10     process 3      finished arr    4      total    3      remain    0      wait    3      TA      6 WTA    2.00
At time 10 allocated 32 bytes for process 4 from 0 to 31
At time 10     process 4      started arr   10      total    4      remain    4      wait    0
At time 11 allocated 512 bytes for process 5 from 512 to 1023
At time 11     process 4      stopped arr   10      total    4      remain    3      wait    0
At time 11     process 5      started arr   11      total    2      remain    2      wait    0
At time 13 freed 512 bytes from process 5 from 512 to 1023
At time 13     process 5      finished arr   11      total    2      remain    0      wait    0      TA      2 WTA    1.00
At time 13     process 4      resumed arr   10      total    4      remain    3      wait    2
At time 16 freed 32 bytes from process 4 from 0 to 31
At time 16     process 4      finished arr   10      total    4      remain    0      wait    2      TA      6 WTA    1.50
[SCHEDULER] Current Time is 17
[SCHEDULER] Stopping HPF...
[SCHEDULER] Terminated normally...
[SCHEDULER] Logging output...
CPU utilization = 88.89%
Avg WTA = 1.90
Avg Waiting = 1.6
[SCHEDULER] Freeing resources...
[PROCGEN] Terminating...
[PROCGEN] Terminating...Clock Terminating!...
```

3) RR

```
[SCHEDULER] Starting RR...
At time 1 allocated 512 bytes for process 1 from 0 to 511
At time 1 process 1 started arr 1 total 5 remain 5 wait 0
At time 3 allocated 256 bytes for process 2 from 512 to 767
At time 3 process 1 stopped arr 1 total 5 remain 3 wait 0
At time 3 process 2 started arr 3 total 1 remain 1 wait 0
[SCHEDULER] Failed to allocate memory for process [3]: Memory requested 700
At time 4 freed 256 bytes from process 2 from 512 to 767
At time 4 process 2 finished arr 3 total 1 remain 0 wait 0 TA 1 WTA 1.00
At time 4 process 1 resumed arr 1 total 5 remain 3 wait 1
At time 7 freed 512 bytes from process 1 from 0 to 511
At time 7 process 1 finished arr 1 total 5 remain 0 wait 1 TA 6 WTA 1.20
At time 7 allocated 1024 bytes for process 3 from 0 to 1023
At time 7 process 3 started arr 4 total 3 remain 3 wait 3
[SCHEDULER] Failed to allocate memory for process [4]: Memory requested 31
At time 10 freed 1024 bytes from process 3 from 0 to 1023
At time 10 process 3 finished arr 4 total 3 remain 0 wait 3 TA 6 WTA 2.00
At time 10 allocated 32 bytes for process 4 from 0 to 31
At time 10 process 4 started arr 10 total 4 remain 4 wait 0
At time 11 allocated 512 bytes for process 5 from 512 to 1023
At time 12 process 4 stopped arr 10 total 4 remain 2 wait 0
At time 12 process 5 started arr 11 total 2 remain 2 wait 1
At time 14 freed 512 bytes from process 5 from 512 to 1023
At time 14 process 5 finished arr 11 total 2 remain 0 wait 1 TA 3 WTA 1.50
At time 14 process 4 resumed arr 10 total 4 remain 2 wait 2
At time 16 freed 32 bytes from process 4 from 0 to 31
At time 16 process 4 finished arr 10 total 4 remain 0 wait 2 TA 6 WTA 1.50
[SCHEDULER] Current Time is 17
[SCHEDULER] Stopping RR...
[SCHEDULER] Terminated normally...
[SCHEDULER] Logging output...
CPU utilization = 88.89%
Avg WTA = 1.44
Avg Waiting = 1.4
```

4) MLFQ


```

[CPU] CPU utilization = 87.50%
[CPU] Avg WTA = 1.44
[CPU] Avg Waiting = 1.4
[SCHEDULER] Starting MLFL ...
At time 1 allocated 512 bytes for process 1 from 0 to 511
At time 1 process 1 started arr 1 total 5 remain 5 wait 0
At time 3 allocated 256 bytes for process 2 from 512 to 767
At time 3 process 1 stopped arr 1 total 5 remain 3 wait 0
At time 3 process 2 started arr 3 total 1 remain 1 wait 0
[SCHEDULER] Failed to allocate memory for process [3]: Memory requested 700
At time 4 freed 256 bytes from process 2 from 512 to 767
At time 4 process 2 finished arr 3 total 1 remain 0 wait 0 TA 1 WTA 1.00
At time 4 process 1 resumed arr 1 total 5 remain 3 wait 1
At time 6 process 1 stopped arr 1 total 5 remain 1 wait 0
At time 6 process 1 started arr 1 total 5 remain 1 wait 5
At time 7 freed 512 bytes from process 1 from 0 to 511
At time 7 process 1 finished arr 1 total 5 remain 0 wait 1 TA 6 WTA 1.20
At time 7 allocated 1024 bytes for process 3 from 0 to 1023
At time 7 process 3 started arr 4 total 3 remain 3 wait 3
At time 9 process 3 stopped arr 4 total 3 remain 1 wait 0
At time 9 process 3 started arr 4 total 3 remain 1 wait 5
[SCHEDULER] Failed to allocate memory for process [4]: Memory requested 31
At time 10 freed 1024 bytes from process 3 from 0 to 1023
At time 10 process 3 finished arr 4 total 3 remain 0 wait 3 TA 6 WTA 2.00
At time 10 allocated 32 bytes for process 4 from 0 to 31
At time 10 process 4 started arr 10 total 4 remain 4 wait 0
At time 11 allocated 512 bytes for process 5 from 512 to 1023
At time 12 process 4 stopped arr 10 total 4 remain 2 wait 0
At time 12 process 5 started arr 11 total 2 remain 2 wait 1
At time 14 freed 512 bytes from process 5 from 512 to 1023
At time 14 process 5 finished arr 11 total 2 remain 0 wait 1 TA 3 WTA 1.50
At time 14 process 4 started arr 10 total 4 remain 2 wait 4
At time 16 freed 32 bytes from process 4 from 0 to 31
At time 16 process 4 finished arr 10 total 4 remain 0 wait 2 TA 6 WTA 1.50
[SCHEDULER] Current Time is 17
[SCHEDULER] Stopping MLFL...
[SCHEDULER] Terminated normally...
[SCHEDULER] Logging output...
CPU utilization = 87.50%
Avg WTA = 1.44
Avg Waiting = 1.4
[SCHEDULER] Freeing resources...
[SCHEDULER] Freeing resources...Clock Terminating!...[PROCGEN] Terminating...

```

Workload Distribution

Hereby an excel sheet link of the workload distribution for phase 1 and 2:

 OS Project To-do list