OS Project Report

Team 14

Name	Sec.	BN.
Ahmed Magdy Ahmed	1	5
Mohamed Ahmed AbdelMonssef	2	14
Menna Mahmoud	2	26
Nada Abdelmaboud	2	27

1.Phase 1:

1.1 Data-structures:

1.1.1 2-Level Priority Queue:

- First level: each node represents a linked list that's is sorted with the priority.
- Second level: each linked list in the first level is a list of all processes have the same priority sorted by the arrival time.

1.1.2 Circular Queue:

- It is a regular queue, where each node is connected to its next. The name "Circular" demonstrates the way we are dealing with it. We pop a process, then give it its quantum then push it at the end of the queue.
- Optimizes the round robin algorithm as each insert – select – delete is O(1).

1.2 IPCs:

1. **Process generator and Scheduler:**
    a. SIGUSR1 is sent from the scheduler to the generator to inform it that the scheduler has started.
    b. SIGUSR1 is sent from the generator to the scheduler to notify the scheduler that all processes are sent.
    c. A message queue is used to send processes from the generator to the scheduler.
    d. A semaphore is used to synchronize the processes sending from the generator to the scheduler. The scheduler won't start receiving new processes until the generator sends all the processes of the current clk.
2. **Scheduler and processes:**
    a. A shared memory is used to make the scheduler able to read the remining time of the running process.
    b. A semaphore is used to ensure that the scheduler won't read the remaining time before the process updates it.
    c. SIGSTOP is sent from the scheduler to the process to simulate it has got back to the ready queue.
    d. SIGCONT is sent from the scheduler to the process to simulate it has got back to the running state.
    e. SIGUSR1 is sent from the process to the scheduler to notify the scheduler that it finished running.

1.3 Methodology:

- process generator

1. Generator reads the input file.
2. Algorithm type is read by scanf (+ quantum in case of round robin).
3. Generator creates (forks) the clk and scheduler processes.
4. When scheduler is ready it sends a signal to generator.
5. When generator receives a signal from scheduler it starts sending the processes.
6. Each second all processes arrived in this second are sent from the generator to the scheduler through a message queue, then the generator ups the semaphore.
7. When all processes are sent, the generator sends a signal to the scheduler to notify it.
8. The generator waits for the scheduler exit, then terminates all resources and exits.

- scheduler

1. Scheduler gets the algorithm and number of processes and quantum, in Round Robin case, as parameters from generator then checks for the algorithm type.
2. While number of processes is not zero it receives all processes in the current second and perform the scheduling algorithm.
3. When a process is received it's inserted in the PCB array which contains the processes data.
4. Scheduling Algorithms:
    - 4.1 HPF (Highest Priority First):
        - * The new process is pushed to the priority queue.
        - * If there isn't a process running it creates the new process and send the run time in its parameters.
        - * If there is a process running the new process waits in the queue till its turn.
    - 4.2 SRTN (Shortest Remaining Time Next):
        - * The new process is pushed to the priority queue
        - * If there isn't a process running it creates the new process and send the remaining time in its parameters
        - * If there is a process running the scheduler reads the remaining time of the running process from a shared memory between scheduler and process then compares it with the run time of the new process if more than the new process it stops the current process and forks the new process.
    - 4.3 RR (Round Robin):
        - * The new processes are inserted in the queue.

* Pop a process from the queue and give it a quantum.
* After the running process finishes its quantum, it is pushed to the end of the queue and a new process is extracted from the queue and starts its quantum.

- process

1. Decrements its remaining time each second.
2. Writes its remaining time each second in the shared memory.
3. Notifies the scheduler about its end by sending SIGUSR2 signal to the scheduler.

Assumptions:

1. Processes.txt should have an empty line at the end
2. Processes are sorted by their id and arrival time.

Workload distribution:

Name	Tasks
Ahmed Magdy	**Circular Queue, Generator, Round Robin, Phase 3**
Mohamed Ahmed AbdelMonsef	**IPCS, Round Robin, Phase 3**
Menna Mahmoud	**Priority Queue, SRTN, HPF, Phase 2**
Nada Abdelmaboud	**Priority Queue, SRTN, HPF, Phase 2**

Time Table:

Task	Time
Circular Queue	**2 Hours**
Priority Queue	**1/2 Day**
Generator	**5 Hours**
IPCS	**2 Days**
SRTN	**1 Day**
HPF	**1 Day**
Round Robin	**1 Day**
Phase 2	**2 Days**
Phase 3	**1 Day**

Note: Fixing Bugs took more time than the whole project implementation .

1.4 Testcases(Also added to the attatched test cases folder):

1.4.1 testcase #1:

```
#id arrival runtime priority
1   6   1   2
2   9   23  8
3   14  1   4
4   24  6   7
```

HPF output :

```
At time 6 process 1 started arr 6 total 1 remain 1 wait 0
At time 7 process 1 finished arr 6 total 1 remain 0 wait 0 TA 1 WTA 1
At time 9 process 2 started arr 9 total 23 remain 23 wait 0
At time 32 process 2 finished arr 9 total 23 remain 0 wait 0 TA 23 WTA 1
At time 32 process 3 started arr 14 total 1 remain 1 wait 18
At time 33 process 3 finished arr 14 total 1 remain 0 wait 18 TA 19 WTA 19
At time 33 process 4 started arr 24 total 6 remain 6 wait 9
At time 39 process 4 finished arr 24 total 6 remain 0 wait 9 TA 15 WTA 2.5
```

STFN output:

```
At time 6 process 1 started arr 6 total 1 remain 1 wait 0
At time 7 process 1 finished arr 6 total 1 remain 0 wait 0 TA 1 WTA 1
At time 9 process 2 started arr 9 total 23 remain 23 wait 0
At time 14 process 2 stopped arr 9 total 23 remain 18 wait 0
At time 14 process 3 started arr 14 total 1 remain 1 wait 0
At time 15 process 3 finished arr 14 total 1 remain 0 wait 0 TA 1 WTA 1
At time 15 process 2 resumed arr 9 total 23 remain 18 wait 1
At time 24 process 2 stopped arr 9 total 23 remain 8 wait 1
At time 24 process 4 started arr 24 total 6 remain 6 wait 0
At time 30 process 4 finished arr 24 total 6 remain 0 wait 0 TA 6 WTA 1
At time 30 process 2 resumed arr 9 total 23 remain 8 wait 7
At time 38 process 2 finished arr 9 total 23 remain 0 wait 7 TA 29 WTA 1.26
```

RR output(Quantum = 4) :

```
At time 6 process 1 started arr 6 total 1 remain 1 wait 0
At time 7 process 1 finished arr 6 total 1 remain 0 wait 0 TA 1 WTA 1
At time 9 process 2 started arr 9 total 23 remain 23 wait 0
At time 13 process 2 stopped arr 9 total 23 remain 19 wait 0
At time 13 process 2 resumed arr 9 total 23 remain 19 wait 0
At time 17 process 2 stopped arr 9 total 23 remain 15 wait 0
At time 17 process 3 started arr 14 total 1 remain 1 wait 3
At time 18 process 3 finished arr 14 total 1 remain 0 wait 3 TA 4 WTA 4
At time 18 process 2 resumed arr 9 total 23 remain 15 wait 4
At time 22 process 2 stopped arr 9 total 23 remain 10 wait 4
At time 22 process 2 resumed arr 9 total 23 remain 10 wait 4
At time 26 process 2 stopped arr 9 total 23 remain 6 wait 4
At time 26 process 4 started arr 24 total 6 remain 6 wait 2
At time 30 process 4 stopped arr 24 total 6 remain 2 wait 2
At time 30 process 2 resumed arr 9 total 23 remain 6 wait 8
At time 34 process 2 stopped arr 9 total 23 remain 2 wait 8
At time 34 process 4 resumed arr 24 total 6 remain 2 wait 6
At time 36 process 4 finished arr 24 total 6 remain 0 wait 6 TA 12 WTA 2
At time 37 process 2 resumed arr 9 total 23 remain 2 wait 15
At time 39 process 2 finished arr 9 total 23 remain 0 wait 15 TA 30 WTA 1.3
```

1.4.1 testcase #2:

```
#id arrival runtime priority
1   5   17  10
2   5   24  2
3   14  26  2
4   17  20  6
5   20  10  3
```

HPF output :

```
At time 5 process 2 started arr 5 total 24 remain 24 wait 0
At time 29 process 2 finished arr 5 total 24 remain 0 wait 0 TA 24 WTA 1
At time 29 process 3 started arr 14 total 26 remain 26 wait 15
At time 55 process 3 finished arr 14 total 26 remain 0 wait 15 TA 41 WTA 1.58
At time 55 process 5 started arr 20 total 10 remain 10 wait 35
At time 65 process 5 finished arr 20 total 10 remain 0 wait 35 TA 45 WTA 4.5
At time 65 process 4 started arr 17 total 20 remain 20 wait 48
At time 85 process 4 finished arr 17 total 20 remain 0 wait 48 TA 68 WTA 3.4
At time 85 process 1 started arr 5 total 17 remain 17 wait 80
At time 102 process 1 finished arr 5 total 17 remain 0 wait 80 TA 97 WTA 5.71
```

STFN output :

```
At time 5 process 1 started arr 5 total 17 remain 17 wait 0
At time 22 process 1 finished arr 5 total 17 remain 0 wait 0 TA 17 WTA 1
At time 22 process 5 started arr 20 total 10 remain 10 wait 2
At time 32 process 5 finished arr 20 total 10 remain 0 wait 2 TA 12 WTA 1.2
At time 32 process 4 started arr 17 total 20 remain 20 wait 15
At time 52 process 4 finished arr 17 total 20 remain 0 wait 15 TA 35 WTA 1.75
At time 52 process 2 started arr 5 total 24 remain 24 wait 47
At time 76 process 2 finished arr 5 total 24 remain 0 wait 47 TA 71 WTA 2.96
At time 76 process 3 started arr 14 total 26 remain 26 wait 62
At time 102 process 3 finished arr 14 total 26 remain 0 wait 62 TA 88 WTA 3.38
```

RR output(Quantum = 10) :

```
At time 5 process 1 started arr 5 total 17 remain 17 wait 0      You, seconds ago
At time 15 process 1 stopped arr 5 total 17 remain 7 wait 0
At time 15 process 2 started arr 5 total 24 remain 24 wait 10
At time 25 process 2 stopped arr 5 total 24 remain 14 wait 10
At time 25 process 3 started arr 14 total 26 remain 26 wait 11
At time 35 process 3 stopped arr 14 total 26 remain 16 wait 11
At time 35 process 1 resumed arr 5 total 17 remain 7 wait 20
At time 42 process 1 finished arr 5 total 17 remain 0 wait 20 TA 37 WTA 2.18
At time 42 process 4 started arr 17 total 20 remain 20 wait 25
At time 52 process 4 stopped arr 17 total 20 remain 10 wait 25
At time 52 process 5 started arr 20 total 10 remain 10 wait 32
At time 62 process 5 finished arr 20 total 10 remain 0 wait 32 TA 42 WTA 4.19
At time 62 process 2 resumed arr 5 total 24 remain 14 wait 52
At time 72 process 2 stopped arr 5 total 24 remain 4 wait 52
At time 72 process 3 resumed arr 14 total 26 remain 16 wait 48
At time 82 process 3 stopped arr 14 total 26 remain 6 wait 48
At time 82 process 4 resumed arr 17 total 20 remain 10 wait 55
At time 92 process 4 finished arr 17 total 20 remain 0 wait 55 TA 75 WTA 3.75
At time 92 process 2 resumed arr 5 total 24 remain 4 wait 92
At time 96 process 2 finished arr 5 total 24 remain 0 wait 92 TA 91 WTA 3.79
At time 96 process 3 resumed arr 14 total 26 remain 6 wait 72
At time 102 process 3 finished arr 14 total 26 remain 0 wait 72 TA 88 WTA 3.38
```

1.4.1 testcase #3

```
#id arrival runtime priority
1   7   10  2
2   9   5   8
3   19  9   5
4   22  2   9
5   25  10  9
6   29  18  1
7   39  8   2
```

HPF output :

```
At time 7 process 1 started arr 7 total 10 remain 10 wait 0
At time 17 process 1 finished arr 7 total 10 remain 0 wait 0 TA 10 WTA 1
At time 17 process 2 started arr 9 total 5 remain 5 wait 8
At time 22 process 2 finished arr 9 total 5 remain 0 wait 8 TA 13 WTA 2.6
At time 22 process 3 started arr 19 total 9 remain 9 wait 3
At time 31 process 3 finished arr 19 total 9 remain 0 wait 3 TA 12 WTA 1.33
At time 31 process 6 started arr 29 total 18 remain 18 wait 2
At time 49 process 6 finished arr 29 total 18 remain 0 wait 2 TA 20 WTA 1.11
At time 49 process 7 started arr 39 total 8 remain 8 wait 10
At time 57 process 7 finished arr 39 total 8 remain 0 wait 10 TA 18 WTA 2.25
At time 57 process 4 started arr 22 total 2 remain 2 wait 35
At time 59 process 4 finished arr 22 total 2 remain 0 wait 35 TA 37 WTA 18.5
At time 59 process 5 started arr 25 total 10 remain 10 wait 34
At time 69 process 5 finished arr 25 total 10 remain 0 wait 34 TA 44 WTA 4.4
```

STFN output:

```
At time 7 process 1 started arr 7 total 10 remain 10 wait 0      You, 11 minute
At time 9 process 1 stopped arr 7 total 10 remain 8 wait 0
At time 9 process 2 started arr 9 total 5 remain 5 wait 0
At time 14 process 2 finished arr 9 total 5 remain 0 wait 0 TA 5 WTA 1
At time 14 process 1 resumed arr 7 total 10 remain 8 wait 5
At time 22 process 1 finished arr 7 total 10 remain 0 wait 5 TA 15 WTA 1.5
At time 22 process 4 started arr 22 total 2 remain 2 wait 0
At time 24 process 4 finished arr 22 total 2 remain 0 wait 0 TA 2 WTA 1
At time 24 process 3 started arr 19 total 9 remain 9 wait 5
At time 33 process 3 finished arr 19 total 9 remain 0 wait 5 TA 14 WTA 1.56
At time 33 process 5 started arr 25 total 10 remain 10 wait 8
At time 43 process 5 finished arr 25 total 10 remain 0 wait 8 TA 18 WTA 1.8
At time 43 process 7 started arr 39 total 8 remain 8 wait 4
At time 51 process 7 finished arr 39 total 8 remain 0 wait 4 TA 12 WTA 1.5
At time 51 process 6 started arr 29 total 18 remain 18 wait 22
At time 69 process 6 finished arr 29 total 18 remain 0 wait 22 TA 40 WTA 2.22
```

RR output(Quantum = 10) :

```
At time 7 process 1 started arr 7 total 10 remain 10 wait 0      You, seconds ago
At time 17 process 1 finished arr 7 total 10 remain 0 wait 0 TA 10 WTA 1
At time 17 process 2 started arr 9 total 5 remain 5 wait 8
At time 22 process 2 finished arr 9 total 5 remain 0 wait 8 TA 13 WTA 2.6
At time 22 process 3 started arr 19 total 9 remain 9 wait 3
At time 31 process 3 finished arr 19 total 9 remain 0 wait 3 TA 12 WTA 1.33
At time 31 process 4 started arr 22 total 2 remain 2 wait 9
At time 33 process 4 finished arr 22 total 2 remain 0 wait 9 TA 11 WTA 5.5
At time 33 process 5 started arr 25 total 10 remain 10 wait 8
At time 43 process 5 finished arr 25 total 10 remain 0 wait 8 TA 18 WTA 1.8
At time 43 process 6 started arr 29 total 18 remain 18 wait 14
At time 53 process 6 stopped arr 29 total 18 remain 8 wait 14
At time 53 process 7 started arr 39 total 8 remain 8 wait 14
At time 61 process 7 finished arr 39 total 8 remain 0 wait 14 TA 22 WTA 2.75
At time 61 process 6 resumed arr 29 total 18 remain 8 wait 36
At time 69 process 6 finished arr 29 total 18 remain 0 wait 36 TA 40 WTA 2.22
```

2.Phase 2:

2.1 Data-structures :

-Array of linked lists , each list represents the memory blocks of this list size
sizes of lists are multiples of 2 till 2 power 10 (1024 -memory size).
- each list contains blocks and each block has an offset address and the process id .

2.2 Methodology -Buddy Algorithm-:

2.2.1 Allocating new process in memory :

- get the list index of the smallest size of power 2 that fits the process size by ceiling the log2 of process size
- search starting from this list to higher sizes lists for the first free block
- if there is a free block with the smallest fit size of the new process found then update this block with the process data
- if there is a free block with higher size found the block keeps splitting into 2 blocks one is inserted in the higher size list and the other is splitted again till reaching the smallest fit size then a new block is inserted in the smallest fit size list.

2.2.1 Deallocating process in memory :

- get the list index of the smallest size of power 2 that fits the process size by ceiling the log2 of process size
- search for the process id in this list then free the block by making the id= -1
- send the freed block and its neighbor (next if even,prev is odd) to a merge algorithm
- the merge method checks for ability to merge the 2 blocks and if merge is allowed then the 2 blocks are deleted completely from the list and a new block with the size exactly higher than them is added then in a recursive way the merging method keeps merging freed blocks .

2.3 Testcases :

2.3.1 testcase #1

```
#id arrival runtime priority memSize
1   10  14  1   30
2   12  4   7   100
3   20  1   3   1
4   30  28  8   2
5   31  11  6   256
6   31  1   7   100
7   31  1   3   10
```

Output (HPF Scheduling):

```
At time 10 allocated 30 bytes for process 1 from 0 to 31
At time 24 freed 30 bytes for process 1 from 0 to 31
At time 24 allocated 1 bytes for process 3 from 0 to 1
At time 25 freed 1 bytes for process 3 from 0 to 1
At time 25 allocated 100 bytes for process 2 from 0 to 127
At time 29 freed 100 bytes for process 2 from 0 to 127
At time 30 allocated 2 bytes for process 4 from 0 to 1
At time 58 freed 2 bytes for process 4 from 0 to 1
At time 58 allocated 10 bytes for process 7 from 0 to 15
At time 59 freed 10 bytes for process 7 from 0 to 15
At time 59 allocated 256 bytes for process 5 from 0 to 255
At time 70 freed 256 bytes for process 5 from 0 to 255
At time 70 allocated 100 bytes for process 6 from 0 to 127
At time 71 freed 100 bytes for process 6 from 0 to 127
```

2.3.1 testcase #2

```
#id arrival runtime priority memSize
1   2   17  0   10
2   3   10  0   20
3   19  3   8   30
4   22  21  0   5
5   26  12  1   200
```

Output (STNF Scheduling):

```
At time 2 allocated 10 bytes for process 1 from 0 to 15
At time 3 allocated 20 bytes for process 2 from 32 to 63
At time 13 freed 20 bytes for process 2 from 32 to 63
At time 19 allocated 30 bytes for process 3 from 32 to 63
At time 22 freed 30 bytes for process 3 from 32 to 63
At time 32 freed 10 bytes for process 1 from 0 to 15
At time 32 allocated 200 bytes for process 5 from 0 to 255
At time 44 freed 200 bytes for process 5 from 0 to 255
At time 44 allocated 5 bytes for process 4 from 0 to 7
At time 65 freed 5 bytes for process 4 from 0 to 7
```

2.3.1 testcase #3

```
#id arrival runtime priority memSize
1   6   5   0   130
2   9   15  6   60
3   15  2   10  2
4   21  16  3   10
5   21  17  9   55
                        nadaabdelmabo
```

Output (RR Scheduling Where Quantum = 10):

```
At time 6 allocated 130 bytes for process 1 from 0 to 255
At time 11 freed 130 bytes for process 1 from 0 to 255
At time 11 allocated 60 bytes for process 2 from 0 to 63
At time 17 allocated 2 bytes for process 3 from 64 to 65
At time 19 freed 2 bytes for process 3 from 64 to 65
At time 25 allocated 10 bytes for process 4 from 64 to 79
At time 31 allocated 55 bytes for process 5 from 128 to 191
At time 40 freed 60 bytes for process 2 from 0 to 63
At time 56 freed 10 bytes for process 4 from 64 to 79
At time 61 freed 55 bytes for process 5 from 128 to 191
```