Project Report

Team number: **11**

Data structure:

1- Struct PCB:

- For saving the process data.
- It contains process's ID, pid, arrival time, burst time, start time, finish time, remaining time, state, priority, index, wait time, last run time, memory size, memory start index, memory last index.

Function	Description
Compare (PCB, PCB, Sorting key)	Return the difference between two given PCB elements based on the given key which has to be a valid PCB member value.
Swap (PCB, PCB)	Exchange member values of two given PCB elements
Equalize (PCB, PCB)	Given two PCB elements, set them equal to each other
Print (PCB)	Print the given PCB member values

2- Vector of PCB:

- To store list of processes.
- This vector can be used as a normal array or as a queue.

Vector as a queue (used for round robin)

Function	Description
Push (PCB)	Given a process push it at the end of the queue.
Pop ()	Pop a process from the top of the queue.
Top ()	Return the first element of the queue.

Vector as an array (used for HPF and SRTN)

Function	Description
Find (ID)	Given the process's ID, search for the process.
Get (index)	Given the process's index in the array return the process.
Sort (sorting_key)	Generic sorting function used to sort the vector depending on the sorting key. It may be: - Arrival time - Burst time - Start time - Finish time - Last run time - Priority

General usage functions

Function	Description
Initialize (size)	Initialize the vector with given size.
Size (vector)	Return size of the given vector.
printVector(vector)	Print the all vector's elements.

3- Buddy algorithm vector:

- Vector of vector of pairs to store the free available blocks.
- We have 10 slots as the memory size is 1024 = 2^{10} .Each slot contains a vector with the available blocks of size i where i is the index of the slot. Each block is represented as pair (start index, last index).

File Reading Functions:

Function	Description
fileHandler	It reads a test case file named "processes.txt", and fill a vector then returns this vector to be used as process table.
testVectorValidity	It has two parts each to test the vector as an array or as a queue. It tries all the provided member functions in a simple procedure to validate that the data structure and its member functions are working properly.

Algorithms Explanations:

1- Round Robin

- Round Robin with quantum Q, check for every clock cycle if there is a new process arrived and check if it can be allocated in memory then it adds it to a circular queue where it loops on this queue and run each process for max of Q until it finishes and get removed from the Queue, if it has no memory left to be allocated then its added to the last position in the queue so the scheduler can check on it later after some other processes have already left the memory, when a process finish the scheduler wait for it to return and exit, until there are no more processes and the queue is empty then the scheduler clear all IPC used and exit

2- Shortest Remaining Time Next

- Shortest remaining time next algorithm, for every clock check if there is a new coming process with a shorter burst time than the currently running one, if any is running. If there is one with a shorter time, then stop the current running process and move the new coming process to a new vector that contains all processes those got a response and had a time to run, and this new process will start running. If a process finishes its running time, look for the new coming process which has the burst time, and compare between it and the process in the “got-response” vector that has shortest remaining time among that vector, by the compare the one that has a shorter remaining time will be executed. Same cycle is running until both coming processes queue and the got-response queue are empty.

3- Highest Priority First

- Using an array of processes, we sort it w.r.t. priority and then we pick the highest priority process if it’s already arrived and it can be allocated in memory if not, we proceed to the second highest priority etc. until we serve N processes.

Inter-Process Communication

- We use 2 message queues to communicate between process and scheduler, one serves as up queue and other as down queue to make sure that the order of execution is correct.
- We put Process Table in Shared Memory so it is available to any process with the right key.
- Each process run for 1 cycle and then check if it has either finished or stopped or should continue running for another cycle.
- We put the memory array in shared memory (char array[1024]) so it is available to any process with the right key.

Assumptions:

- The test case file **must** be in the same place with "process_generator.c" when running the project.
- No process with run time less than 1 as it doesn’t make sense to have a process with running time of zero
- No process with memory size less than zero.

- In highest priority first algorithm if two processes have the same priority, we pick any of them
- Utilization is the running time of all processes over total time of the scheduler running time.

$$\frac{\sum_{i=0}^{N} burst_{time}[i]}{totalrunningtime}$$

Task	Time
Implementing algorithms (RR – SRTN – HPF)	1 day – 3 days
Data structure	3 days
Implementing process generator	1 day
Input file reading	1 day
Design Communication system	3 days
Implementing buddy algorithm	1 day
Input file reading and Edit process generator	1 day
Add PCB's data members and initialize shared memory	1 day
Integration for 2 phases	2 day
Fix bugs for 2 phases	3 day

Eman Othman	- HPF algorithm - Integrate with scheduler.c - Integrate process generator - HPF algorithm integration with buddy algorithm. - Buddy algorithm (allocate – deallocate – merge buddies blocks)

Bahaa Eldeen Mohamed	- Round robin algorithm - Round robin algorithm integration with buddy algorithm. - Integrate with scheduler.c - Integrate process generator - IPC (message queues, shared memory) - Memory management.
Tarek Samy	- Data structure. - Input file reading - Implement process generator - Integrate with scheduler - Initialize shared memory.
Abdel Rahman Fadl	- SRT algorithm. - SRT algorithm integration with buddy algorithm. - Integrate with scheduler.c - Integrate process generator.

Results:

Phase 1

Testcase #1:

Input

```
#id arrival runtime priority
1       6       9       4
2       9       2       4
3       13      6       1
4       20      13      6
```

Round Robin q=5 scheduler.log

```
#At time x process y state arr w total z remain y wait k
At time 6 process 1 started arr 6 total 9 remain 9 wait 0
At time 11 process 1 stopped arr 6 total 9 remain 4 wait 0
At time 11 process 2 started arr 9 total 2 remain 2 wait 2
At time 13 process 2 finished arr 9 total 2 remain 0 wait 2 TA 4 WTA 2.00
At time 13 process 1 resumed arr 6 total 9 remain 4 wait 2
At time 17 process 1 finished arr 6 total 9 remain 0 wait 2 TA 11 WTA 1.22
At time 17 process 3 started arr 13 total 6 remain 6 wait 4
At time 22 process 3 stopped arr 13 total 6 remain 1 wait 4
At time 22 process 4 started arr 20 total 13 remain 13 wait 2
At time 27 process 4 stopped arr 20 total 13 remain 8 wait 2
At time 27 process 3 resumed arr 13 total 6 remain 1 wait 9
At time 28 process 3 finished arr 13 total 6 remain 0 wait 9 TA 15 WTA 2.50
At time 28 process 4 resumed arr 20 total 13 remain 8 wait 3
At time 33 process 4 stopped arr 20 total 13 remain 3 wait 3
At time 33 process 4 resumed arr 20 total 13 remain 3 wait 3
At time 36 process 4 finished arr 20 total 13 remain 0 wait 3 TA 16 WTA 1.23
```

Round Robin scheduler.perf

```
CPU utilization = 85.71%
Avg WTA = 1.74
Avg Waiting = 4.00
std WTA = 1.53
```

Shortest remaining time next scheduler.log

```
#At time x process y state arr w total z remain y wait k
At time 6 process 1 started arr 6 total 9 remain 9 wait 0
At time 9 process 1 stopped arr 6 total 9 remain 6 wait 0
At time 9 process 2 started arr 9 total 2 remain 2 wait 0
At time 11 process 2 finished arr 9 total 2 remain 0 wait 0 TA 2 WTA 1.00
At time 11 process 1 resumed arr 6 total 9 remain 6 wait 2
At time 17 process 1 finished arr 6 total 9 remain 0 wait 2 TA 11 WTA 1.22
At time 17 process 3 started arr 13 total 6 remain 6 wait 4
At time 23 process 3 finished arr 13 total 6 remain 0 wait 4 TA 10 WTA 1.67
At time 23 process 4 started arr 20 total 13 remain 13 wait 3
At time 36 process 4 finished arr 20 total 13 remain 0 wait 3 TA 16 WTA 1.23
```

SRTN scheduler.perf

```
CPU utilization = 83.33%
Avg WTA = 1.00
Avg Waiting = 2.25
std WTA = 0.00
```

Highest priority first scheduler.log

```
#At time x process y state arr w total z remain y wait k

At time 6 process 1 started arr 6 total 9 remain 9 wait 0

At time 15 process 1 finished arr 6 total 9 remain 0 wait 0 TA 9 WTA 1.00

At time 15 process 3 started arr 13 total 6 remain 6 wait 2

At time 21 process 3 finished arr 13 total 6 remain 0 wait 2 TA 8 WTA 1.33

At time 21 process 2 started arr 9 total 2 remain 2 wait 12

At time 23 process 2 finished arr 9 total 2 remain 0 wait 12 TA 14 WTA 7.00

At time 23 process 4 started arr 20 total 13 remain 13 wait 3

At time 36 process 4 finished arr 20 total 13 remain 0 wait 3 TA 16 WTA 1.23
```

HPF scheduler.perf

```
CPU utilization = 85.71%

Avg WTA = 2.64

Avg Waiting = 4.25

std WTA = 2.52
```

Testcase #2:

Input

```
#id arrival runtime priority
1       7       1       0
2       15      4       4
```

RR scheduler.log q=5

```
#At time x process y state arr w total z remain y wait k
At time 7 process 1 started arr 7 total 1 remain 1 wait 0
At time 8 process 1 finished arr 7 total 1 remain 0 wait 0 TA 1 WTA 1.00
At time 15 process 2 started arr 15 total 4 remain 4 wait 0
At time 19 process 2 finished arr 15 total 4 remain 0 wait 0 TA 4 WTA 1.00
```

RR scheduler.perf

```
CPU utilization = 27.78%
Avg WTA = 1.00
Avg Waiting = 0.00
std WTA = 0.71
```

STRN scheduler.log

```
#At time x process y state arr w total z remain y wait k
At time 7 process 1 started arr 7 total 1 remain 1 wait 0
At time 8 process 1 finished arr 7 total 1 remain 0 wait 0 TA 1 WTA 1.00
At time 15 process 2 started arr 15 total 4 remain 4 wait 0
At time 19 process 2 finished arr 15 total 4 remain 0 wait 0 TA 4 WTA 1.00
```

SRTN scheduler.perf

```
#At time x process y state arr w total z remain y wait k

At time 7 process 1 started arr 7 total 1 remain 1 wait 0

At time 8 process 1 finished arr 7 total 1 remain 0 wait 0 TA 1 WTA 1.00

At time 15 process 2 started arr 15 total 4 remain 4 wait 0

At time 19 process 2 finished arr 15 total 4 remain 0 wait 0 TA 4 WTA 1.00
```

HPF scheduler.log

```
#At time x process y state arr w total z remain y wait k

At time 7 process 1 started arr 7 total 1 remain 1 wait 0

At time 8 process 1 finished arr 7 total 1 remain 0 wait 0 TA 1 WTA 1.00

At time 15 process 2 started arr 15 total 4 remain 4 wait 0

At time 19 process 2 finished arr 15 total 4 remain 0 wait 0 TA 4 WTA 1.00
```

HPF scheduler.perf

```
CPU utilization = 27.78%

Avg WTA = 1.00

Avg Waiting = 0.00

std WTA = 0.00
```

Phase 2

Testcase #1:

Input:

```
#id arrival runtime priority memsize
1       6       2       7       89
2       14      17      7       103
```

Round Robin scheduler.log: Q=5

```
#At time x process y state arr w total z remain y wait k
At time 6 process 1 started arr 6 total 2 remain 2 wait 0
At time 8 process 1 finished arr 6 total 2 remain 0 wait 0 TA 2 WTA 1.00
At time 14 process 2 started arr 14 total 17 remain 17 wait 0
At time 19 process 2 stopped arr 14 total 17 remain 12 wait 0
At time 19 process 2 resumed arr 14 total 17 remain 12 wait 0
At time 24 process 2 stopped arr 14 total 17 remain 7 wait 0
At time 24 process 2 resumed arr 14 total 17 remain 7 wait 0
At time 29 process 2 stopped arr 14 total 17 remain 2 wait 0
At time 29 process 2 resumed arr 14 total 17 remain 2 wait 0
At time 31 process 2 finished arr 14 total 17 remain 0 wait 0 TA 17 WTA 1.00
```

Round Robin scheduler.perf:

```
CPU utilization = 63.33%
Avg WTA = 1.00
Avg Waiting = 0.00
std WTA = 0.71
```

Round Robin memory.log

```
#At time x allocated y bytes for process z form i to j
At time 6 allocated 128 bytes for process 1 from 0 to 127
At time 8 freed 128 bytes from process 1 from 0 to 127
At time 14 allocated 128 bytes for process 2 from 0 to 127
At time 31 freed 128 bytes from process 2 from 0 to 127
```

Shortest remaining time next scheduler.log

```
#At time x process y state arr w total z remain y wait k
At time 6 process 1 started arr 6 total 2 remain 2 wait 0
At time 8 process 1 finished arr 6 total 2 remain 0 wait 0 TA 2 WTA 1.00
At time 14 process 2 started arr 14 total 17 remain 17 wait 0
At time 31 process 2 finished arr 14 total 17 remain 0 wait 0 TA 17 WTA 1.00
```

SRTN scheduler.perf

```
CPU utilization = 59.38%
Avg WTA = 1.00
Avg Waiting = 0.00
std WTA = 0.00
```

SRTN memory .log

```
#At time x allocated y bytes for process z form i to j
At time 6 allocated 128 bytes for process 1 from 0 to 127
At time 8 freed 128 bytes from process 1 from 0 to 127
At time 14 allocated 128 bytes for process 2 from 0 to 127
At time 31 freed 128 bytes from process 2 from 0 to 127
```

Highest Priority First schecduler.log

```
#At time x process y state arr w total z remain y wait k
At time 6 process 1 started arr 6 total 2 remain 2 wait 0
At time 8 process 1 finished arr 6 total 2 remain 0 wait 0 TA 2 WTA 1.00
At time 14 process 2 started arr 14 total 17 remain 17 wait 0
At time 31 process 2 finished arr 14 total 17 remain 0 wait 0 TA 17 WTA 1.00
```

HPF scheduler.perf

```
CPU utilization = 63.33%
Avg WTA = 1.00
Avg Waiting = 0.00
std WTA = 0.00
```

HPF memory.log

```
#At time x allocated y bytes for process z form i to j
At time 6 allocated 128 bytes for process 1 from 0 to 127
At time 8 freed 128 bytes from process 1 from 0 to 127
At time 14 allocated 128 bytes for process 2 from 0 to 127
At time 31 freed 128 bytes from process 2 from 0 to 127
```

Testcase #2

Input:

```
#id arrival runtime priority memsize
1     10     1     4     39
2     11     2     9     211
3     14     3     2     181
```

Round Robin scheduler.log q=7

```
#At time x process y state arr w total z remain y wait k

At time 10 process 1 started arr 10 total 1 remain 1 wait 0

At time 11 process 1 finished arr 10 total 1 remain 0 wait 0 TA 1 WTA 1.00

At time 11 process 2 started arr 11 total 2 remain 2 wait 0

At time 13 process 2 finished arr 11 total 2 remain 0 wait 0 TA 2 WTA 1.00

At time 14 process 3 started arr 14 total 3 remain 3 wait 0

At time 17 process 3 finished arr 14 total 3 remain 0 wait 0 TA 3 WTA 1.00
```

Round Robin scheduler.perf

```
CPU utilization = 37.50%

Avg WTA = 1.00

Avg Waiting = 0.00

std WTA = 0.82
```

Round Robin memory.log

```
#At time x allocated y bytes for process z form i to j

At time 10 allocated 64 bytes for process 1 from 0 to 63

At time 11 freed 64 bytes from process 1 from 0 to 63

At time 11 allocated 256 bytes for process 2 from 0 to 255

At time 13 freed 256 bytes from process 2 from 0 to 255

At time 14 allocated 256 bytes for process 3 from 0 to 255

At time 17 freed 256 bytes from process 3 from 0 to 255
```

SRTN scheduler.log

```
#At time x process y state arr w total z remain y wait k
At time 10 process 1 started arr 10 total 1 remain 1 wait 0
At time 11 process 1 finished arr 10 total 1 remain 0 wait 0 TA 1 WTA 1.00
At time 11 process 2 started arr 11 total 2 remain 2 wait 0
At time 13 process 2 finished arr 11 total 2 remain 0 wait 0 TA 2 WTA 1.00
At time 14 process 3 started arr 14 total 3 remain 3 wait 0
At time 17 process 3 finished arr 14 total 3 remain 0 wait 0 TA 3 WTA 1.00
```

SRTN scheduler.perf

```
CPU utilization = 35.29%
Avg WTA = 1.00
Avg Waiting = 0.00
std WTA = 0.00
```

SRTN memory.log

```
#At time x allocated y bytes for process z from i to j
At time 10 allocated 64 bytes for process 1 from 0 to 63
At time 11 freed 64 bytes from process 1 from 0 to 63
At time 11 allocated 256 bytes for process 2 from 0 to 255
At time 13 freed 256 bytes from process 2 from 0 to 255
At time 14 allocated 256 bytes for process 3 from 0 to 255
At time 17 freed 256 bytes from process 3 from 0 to 255
```

HPF scheduler.log

```
#At time x process y state arr w total z remain y wait k
At time 10 process 1 started arr 10 total 1 remain 1 wait 0
At time 11 process 1 finished arr 10 total 1 remain 0 wait 0 TA 1 WTA 1.00
At time 11 process 2 started arr 11 total 2 remain 2 wait 0
At time 13 process 2 finished arr 11 total 2 remain 0 wait 0 TA 2 WTA 1.00
At time 14 process 3 started arr 14 total 3 remain 3 wait 0
At time 17 process 3 finished arr 14 total 3 remain 0 wait 0 TA 3 WTA 1.00
```

HPF scheduler.perf

```
CPU utilization = 37.50%
Avg WTA = 1.00
Avg Waiting = 0.00
std WTA = 0.00
```

HPF memory.log

```
#At time x allocated y bytes for process z from i to j
At time 10 allocated 64 bytes for process 1 from 0 to 63
At time 11 freed 64 bytes from process 1 from 0 to 63
At time 11 allocated 256 bytes for process 2 from 0 to 255
At time 13 freed 256 bytes from process 2 from 0 to 255
At time 14 allocated 256 bytes for process 3 from 0 to 255
At time 17 freed 256 bytes from process 3 from 0 to 255
```