OS Project Report

Custom Operation System
Team Number: 3

Ahmed Yahia	ID: 55-3969	T-17
Mohamed Elsawy	ID: 55-5566	T-19
Mostafa Hamdy	ID: 55-2962	T-15
Rasheed Atia	ID: 55-4451	T-20
Seifeldin Khaled	ID: 55-25218	T-15
Yousef Yasser	ID: 55-3437	T-15

20 May 2024

1 Aim

The aim of this project is to implement an operating system and analyze how it manages resources and processes. Key components to be implemented include the scheduler, memory management, and synchronization. The scheduler should be capable of efficiently scheduling processes within the system. The scheduler will prioritize and allocate resources to processes based on predefined criteria, optimizing resource utilization and overall system performance. The system implements a memory management system to efficiently allocate and manage resources for process execution. This system will be responsible for storing and organizing the processes within the system.

2 Methodology

In this section, we outline the methodology followed for the development of the simulated operating system (OS), including the implementation of process control blocks (PCB), memory management, scheduler, and mutexes.

2.1 Programs and Program Syntax

Three main programs were provided, each with specific functionalities. Program syntax was defined for various operations such as printing, variable assignment, file operations, and semaphore operations.

2.2 Process Control Block (PCB)

A PCB data structure was designed to store essential information about each process, including Process ID, Process State, Program Counter, and Memory Boundaries. PCBs are crucial for process management and scheduling.

2.3 Memory Management

A memory management system was implemented to allocate memory space for processes. The main memory, consisting of 60 memory words, was divided to accommodate program lines, variables, and PCBs for each process. Each process was assigned sufficient space for instructions and variables.

2.4 Scheduler

A round-robin scheduling algorithm was implemented to schedule processes in the ready queue. one ready queue were used to manage processes waiting to be executed. The scheduler ensured fair execution of processes by rotating them in a round-robin fashion.

- **Round-Robin Scheduling Algorithm:** This algorithm allocates CPU time to each thread or process in a cyclic manner, giving each thread a fixed time slice (quantum) to execute before moving on to the next thread in the queue. It ensures fairness and prevents starvation but may suffer from high context-switching overhead.

2.5 Mutexes and Mutual Exclusion

Mutexes were implemented to enforce mutual exclusion over critical resources, such as file access, user input, and screen output. Three mutexes were created, each corresponding to a resource. The semWait and semSignal instructions were used to control access to resources, ensuring that only one process could use a resource at a time.

2.6 Queues

Three ready queues were established to manage processes waiting to be executed (one blocked queue for each resource). Additionally, a general blocked queue was maintained for processes waiting for any resource to become available.

2.7 Functionality of the Code

The main function initializes semaphores, queues, and defines processes. It prompts the user to enter release times for processes. The main loop simulates clock cycles and executes processes. It checks for new process arrivals, selects processes for execution, and executes instructions. Processes will execute instructions, interact with resources, and transition between states. Processes may block if resources are unavailable, and memory is managed to store process information. The program will print queue and memory states at each clock cycle. Execution will continue until all processes finish, and the program will terminate gracefully

3 Results and Discussion

3.1 Simulation Results

3.1.1 Simulated OS Execution

The simulated OS was designed to read and execute provided programs. The system displayed queues after every scheduling event, indicating which process was currently executing, the instruction being executed, and the memory status in a human-readable format. Sample output of some clock cycles is displayed at the end of the report.

3.1.2 Output

The output of the simulated OS was presented in a readable and presentable format, showcasing the scheduling events, process execution order, and memory status at each clock cycle.

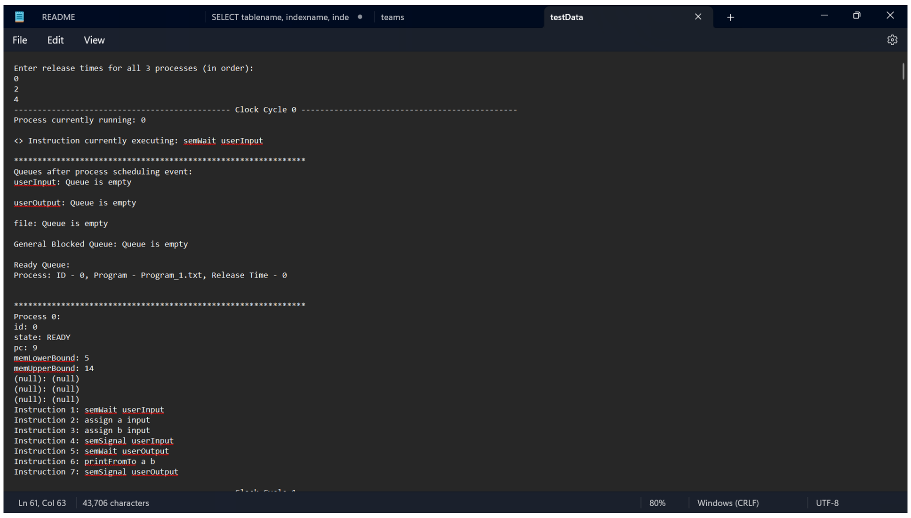
3.2 Discussion

The program's execution provides valuable insights into process scheduling, resource management, and state transitions within the system. Processes dynamically transition between various states, including READY, RUNNING, BLOCKED, and FINISHED, based on their instructions and resource availability. This ensures efficient resource utilization and orderly execution.

Queues play a critical role in managing process readiness and resource allocation. Processes move between the ready queue, indicating their readiness for execution, and various blocked queues, representing their waiting states for specific resources. This enables effective scheduling and prioritization based on dependencies and execution requirements.

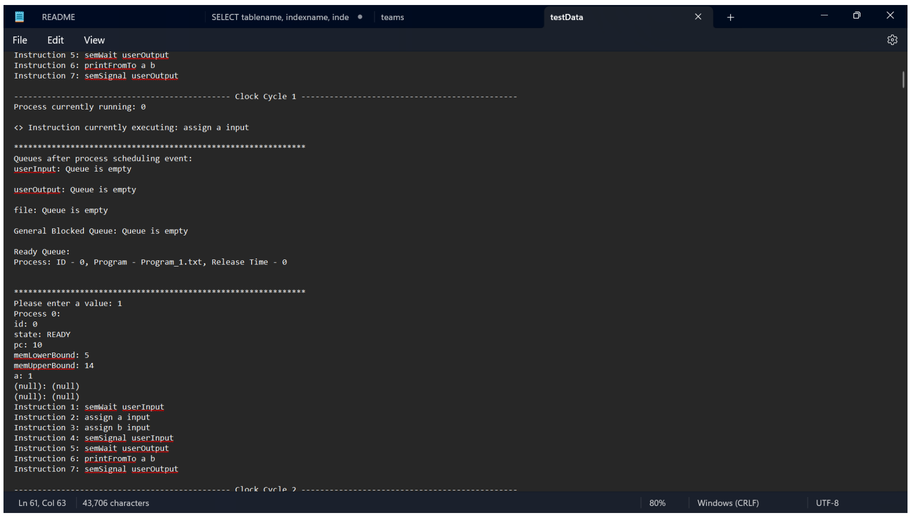
Moreover, semaphore operations such as wait and signal regulate resource access and coordinate concurrent processes. These mechanisms prevent resource contention and deadlock situations, ensuring smooth execution.

In conclusion, the program's execution demonstrates the complexities involved in managing concurrent processes and shared resources within an operating system environment. By effectively coordinating process scheduling, resource allocation, and synchronization, the system optimizes performance and ensures the smooth execution of diverse tasks and operations.



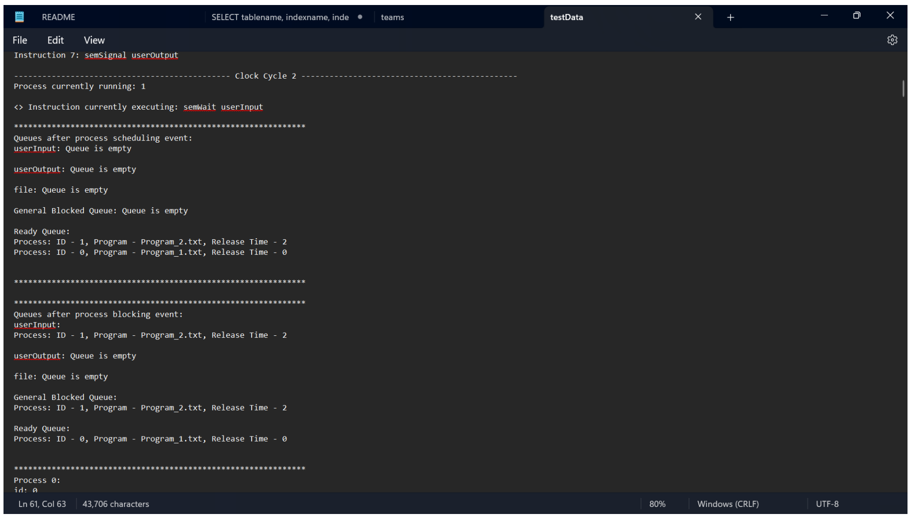
```
Enter release times for all 3 processes (in order):
0
2
4
--------------------------------------------- Clock Cycle 0 ---------------------------------------------
Process currently running: 0

<> Instruction currently executing: semWait userInput

**************************************************************
Queues after process scheduling event:
userInput: Queue is empty

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue: Queue is empty

Ready Queue:
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
Process 0:
id: 0
state: READY
pc: 9
memLowerBound: 5
memUpperBound: 14
(null): (null)
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput
```

(a) Input and Clock Cycle 0



```
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

--------------------------------------------- Clock Cycle 1 ---------------------------------------------
Process currently running: 0

<> Instruction currently executing: assign a input

**************************************************************
Queues after process scheduling event:
userInput: Queue is empty

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue: Queue is empty

Ready Queue:
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
Please enter a value: 1
Process 0:
id: 0
state: READY
pc: 10
memLowerBound: 5
memUpperBound: 14
a: 1
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

--------------------------------------------- Clock Cycle 2 ---------------------------------------------
```

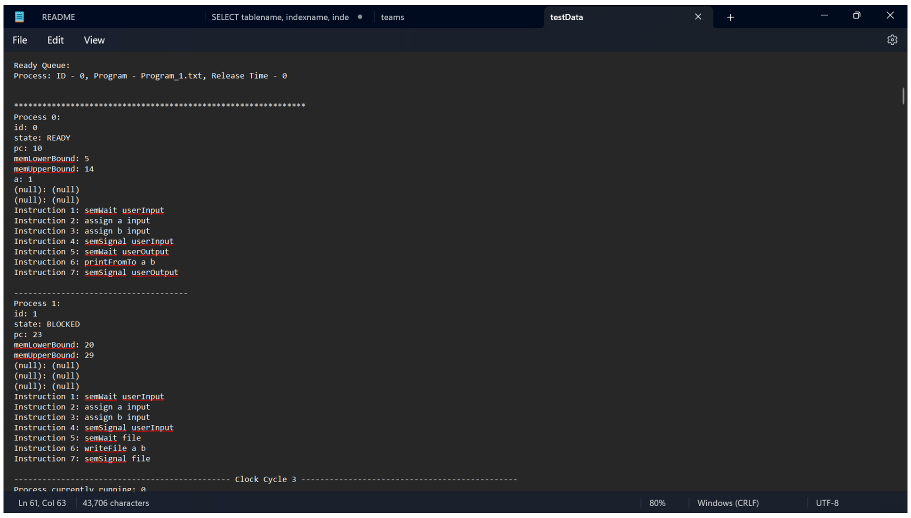
(b) Clock Cycle 1



```
Instruction 7: semSignal userOutput

--------------------------------------------- Clock Cycle 2 ---------------------------------------------
Process currently running: 1

<> Instruction currently executing: semWait userInput

**************************************************************
Queues after process scheduling event:
userInput: Queue is empty

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue: Queue is empty

Ready Queue:
Process: ID - 1, Program - Program_2.txt, Release Time - 2
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************

**************************************************************
Queues after process blocking event:
userInput:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

Ready Queue:
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
Process 0:
id: 0
```

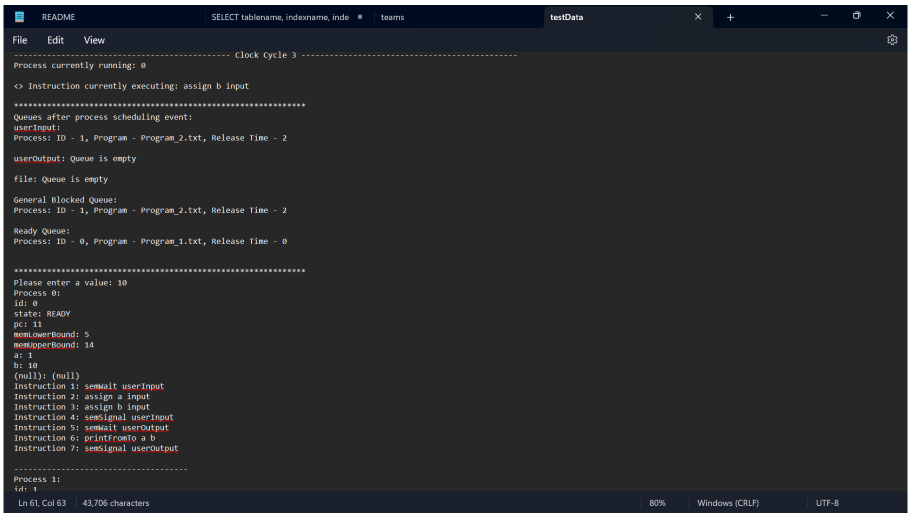
(c) Clock Cycle 2

Figure 1: Simulation Results (Part 1)



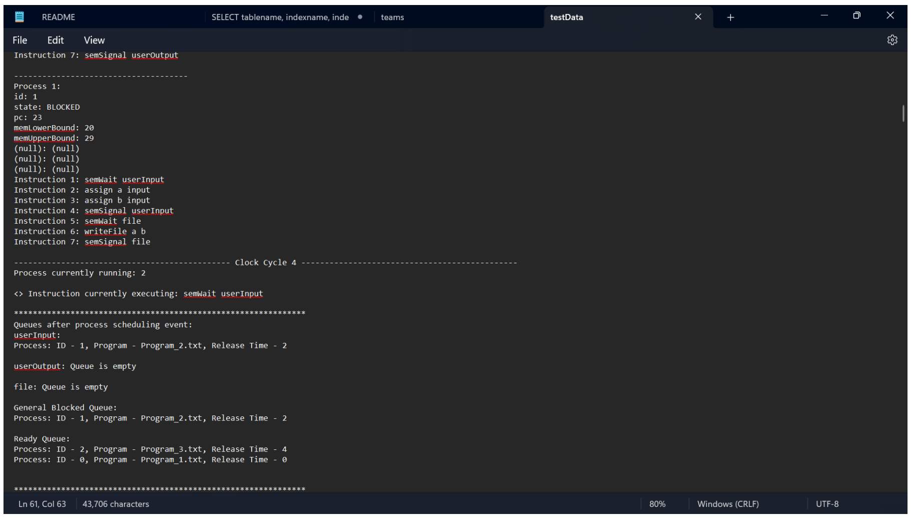
```
Ready Queue:
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
Process 0:
id: 0
state: READY
pc: 10
memLowerBound: 5
memUpperBound: 14
a: 1
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

-------------------------------------
Process 1:
id: 1
state: BLOCKED
pc: 23
memLowerBound: 20
memUpperBound: 29
(null): (null)
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait file
Instruction 6: writeFile a b
Instruction 7: semSignal file

---------------------------------------------- Clock Cycle 3 ----------------------------------------------
Process currently running: 0
```

(a) Clock Cycle 2(2)



```
---------------------------------------------- Clock Cycle 3 ----------------------------------------------
Process currently running: 0

<> Instruction currently executing: assign b input

**************************************************************
Queues after process scheduling event:
userInput:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

Ready Queue:
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
Please enter a value: 10
Process 0:
id: 0
state: READY
pc: 11
memLowerBound: 5
memUpperBound: 14
a: 1
b: 10
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

-------------------------------------
Process 1:
id: 1
```

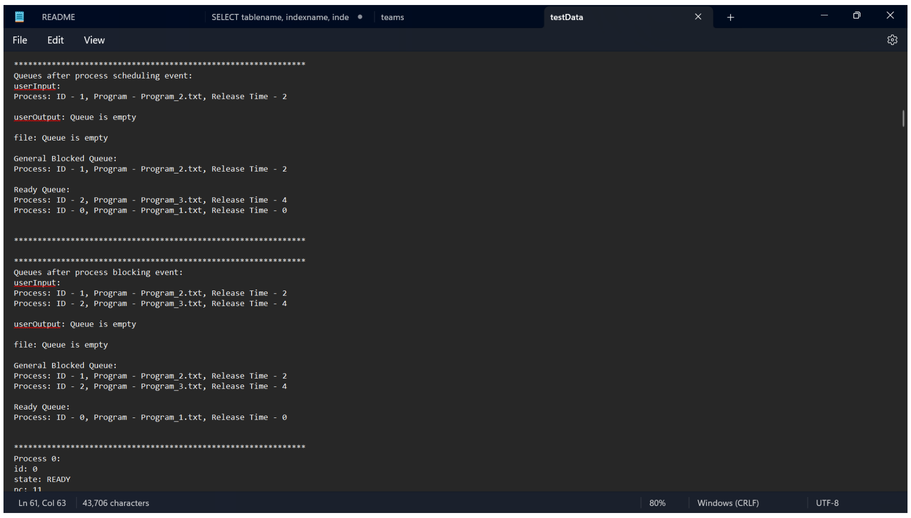
(b) Clock Cycle 3



```
Instruction 7: semSignal userOutput

-------------------------------------
Process 1:
id: 1
state: BLOCKED
pc: 23
memLowerBound: 20
memUpperBound: 29
(null): (null)
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait file
Instruction 6: writeFile a b
Instruction 7: semSignal file

---------------------------------------------- Clock Cycle 4 ----------------------------------------------
Process currently running: 2

<> Instruction currently executing: semWait userInput

**************************************************************
Queues after process scheduling event:
userInput:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

Ready Queue:
Process: ID - 2, Program - Program_3.txt, Release Time - 4
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
```

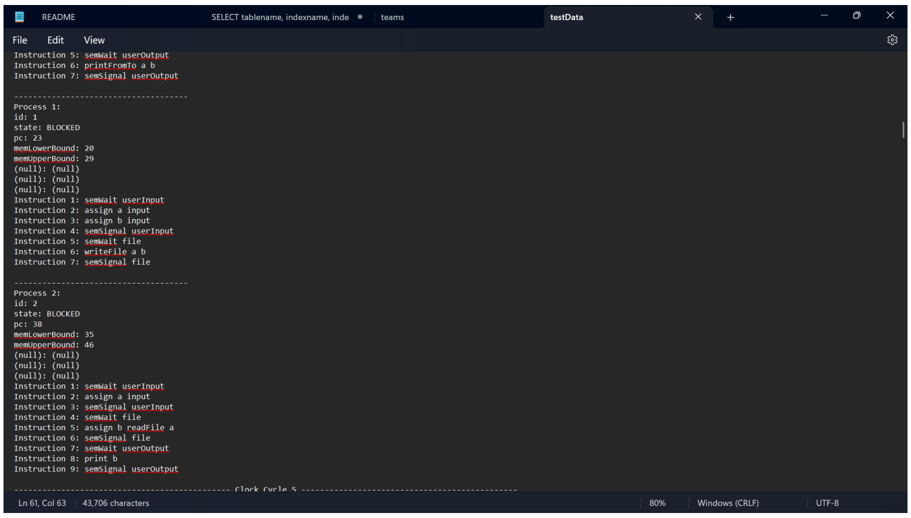
(c) Clock Cycle 3(2)

Figure 2: Simulation Results (Part 2)



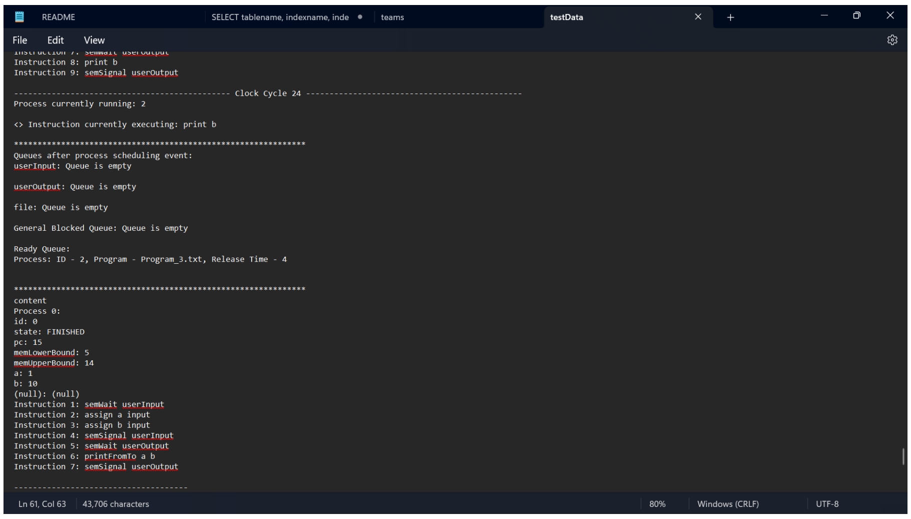
```
**************************************************************
Queues after process scheduling event:
userInput:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue:
Process: ID - 1, Program - Program_2.txt, Release Time - 2

Ready Queue:
Process: ID - 2, Program - Program_3.txt, Release Time - 4
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************

**************************************************************
Queues after process blocking event:
userInput:
Process: ID - 1, Program - Program_2.txt, Release Time - 2
Process: ID - 2, Program - Program_3.txt, Release Time - 4

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue:
Process: ID - 1, Program - Program_2.txt, Release Time - 2
Process: ID - 2, Program - Program_3.txt, Release Time - 4

Ready Queue:
Process: ID - 0, Program - Program_1.txt, Release Time - 0


**************************************************************
Process 0:
id: 0
state: READY
pc: 11
```

(a) Clock Cycle 4



```
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

-------------------------------------
Process 1:
id: 1
state: BLOCKED
pc: 23
memLowerBound: 20
memUpperBound: 29
(null): (null)
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait file
Instruction 6: writeFile a b
Instruction 7: semSignal file

-------------------------------------
Process 2:
id: 2
state: BLOCKED
pc: 38
memLowerBound: 35
memUpperBound: 46
(null): (null)
(null): (null)
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: semSignal userInput
Instruction 4: semWait file
Instruction 5: assign b readFile a
Instruction 6: semSignal file
Instruction 7: semWait userOutput
Instruction 8: print b
Instruction 9: semSignal userOutput

---------------------------------------------- Clock Cycle 5 ----------------------------------------------
```

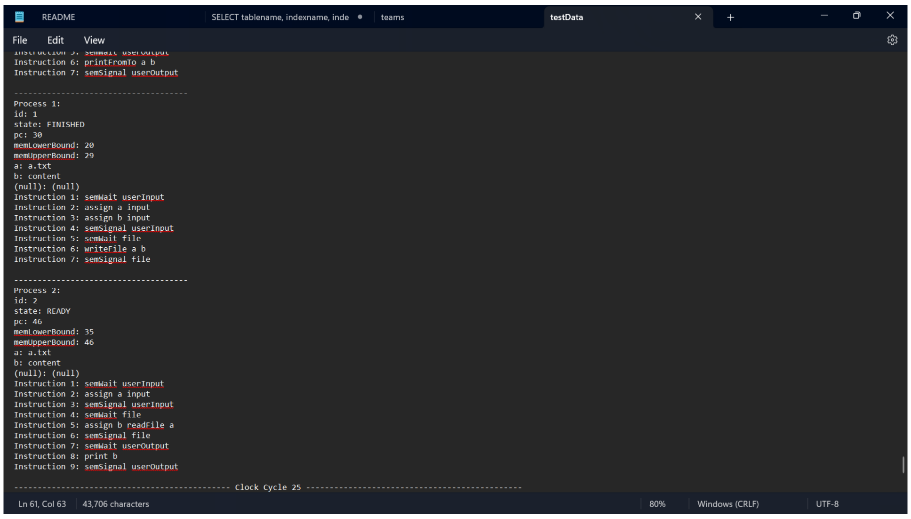
(b) Clock Cycle 4(2)



```
Instruction 7: semWait userOutput
Instruction 8: print b
Instruction 9: semSignal userOutput

---------------------------------------------- Clock Cycle 24 ----------------------------------------------
Process currently running: 2

<> Instruction currently executing: print b

**************************************************************
Queues after process scheduling event:
userInput: Queue is empty

userOutput: Queue is empty

file: Queue is empty

General Blocked Queue: Queue is empty

Ready Queue:
Process: ID - 2, Program - Program_3.txt, Release Time - 4


**************************************************************
content
Process 0:
id: 0
state: FINISHED
pc: 15
memLowerBound: 5
memUpperBound: 14
a: 1
b: 10
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

-------------------------------------
```

(c) Clock Cycle 24

Figure 3: Simulation Results (Part 3)



```
Instruction 5: semWait userOutput
Instruction 6: printFromTo a b
Instruction 7: semSignal userOutput

------------------------------------
Process 1:
id: 1
state: FINISHED
pc: 30
memLowerBound: 20
memUpperBound: 29
a: a.txt
b: content
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: assign b input
Instruction 4: semSignal userInput
Instruction 5: semWait file
Instruction 6: writeFile a b
Instruction 7: semSignal file

------------------------------------
Process 2:
id: 2
state: READY
pc: 46
memLowerBound: 35
memUpperBound: 46
a: a.txt
b: content
(null): (null)
Instruction 1: semWait userInput
Instruction 2: assign a input
Instruction 3: semSignal userInput
Instruction 4: semWait file
Instruction 5: assign b readFile a
Instruction 6: semSignal file
Instruction 7: semWait userOutput
Instruction 8: print b
Instruction 9: semSignal userOutput

--------------------------------------------- Clock Cycle 25 ---------------------------------------------
```

Figure 4: Clock Cycle 24(2)