|  |  |  |
| --- | --- | --- |
| 1. STUDENT INFORMATION | | |
| Name | ID | Section |
| MD SAMIM | 443814935 | 1077 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. PROJECT INFORMATION | | | | | |
| **Title** | implements a message passing scheme in a ring among multiple processes and a distributed mutual exclusion algorithm | | | | |
| Code Links | | | | | Qrcode QR Code Vector Icon. Special Identity Illustration As Simple Sign and ... |
| Source Code | https:// | | | |  |
| Executable Code | https:// | | | |  |
| Approximation of total number of hours spent to complete the project | | | | 5 | |
| Project Due Date | | **24 / 04 / 2025** | Submission Date | / 04 / 2025 | |
|  | | | | | |

|  |
| --- |
| 1. PERSONAL REFLECTION & Self-ASSESSMENT |
| Summarize the project idea and changes you added to the project: |
| The project is about message passing interface and a token base mutual exclusion algorithm in parallel distributed system .I have change all the variable and printing messages and process numbers to reasonable format. |
| After finishing, any New Concepts learned |
| I learned about Microsoft MPI and c programming language. |
| Challenges Faced and Solutions, especially if you submit late: |
| Installing MPI and also setting the paths in environment variable .Overcome errors while compiling and executing the programs |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. Important TESTING AND DEBUGGING Cases | | | | | | |
| # | Test Case / Error | Correct Output | Actual Output | Date/Time | Passed? | Note |
| Problem 2 | | | | | | |
| 1 | Compile program 2 | Should compiled | Compiler not found | 11/04/2025 | Fail | Check the error |
| 2 | Set environment variable | compile | compiled | 11/04/2025 | Pass |  |
| 3 | process sends a message to its successor | Sent correctly | Sent correctly | 11/04/2025 | Pass |  |
| 4 | process receives a message from its predecessor | Received correctly | Received correctly | 11/04/2025 | Pass |  |
| Problem 3 | | | | | | |
| 5 | Run problem 3 | Python not found | Should execute | 12/04/2025 | Fail | Intall python |
| 6 | process can request access to the printer by sending a request message | Send request message | sending a request message | 12/04/2025 | Pass |  |
| 7 | once a process is granted permission to print | print a message indicating that it is printing. | print a message indicating that it is printing. | 12/04/2025 | Pass |  |
| 8 | after printing, the process releases the printer so that other processes can request access. | Released | Released | 12/04/2025 | Pass |  |

|  |  |
| --- | --- |
| 1. COLLABORATION AND RESOURCES | |
| If you received help from any classmates, please provide their names and IDs. This information won't be used against you, but will help in cases where reports are similar. It won't be considered fraud if accompanied by explanations. | |
|  | |
| If you utilized any external tools (e.g., AI, websites), please specify | |
| Name of the tool | Nature of assistance received |
| 1. Geeks For Geeks website | To understand the problem |
| 2.Chatgpt | Asked the steps to solve the problem |
| 3. Microsoft MPI | To create the message passing interface |
| 4. Visual studio code editor | Used for smooth coding and execution |
| 5.Python IDLE | To execute problem 2 python code |
| Steps taken to understand and integrate it into your learning | |
| Firstread the problem and clearly understand the theoretical part then using the tools implemented | |

**ATTACH THE RESULT OF THE PROJECT WITH CODE.**

Problem 2:

#include <mpi.h>  
#include <stdio.h>  
#include <stdlib.h>  
  
int main(int argc, char \*argv[]) {  
 int rank, size;  
 int message;  
 int send\_to, recv\_from;  
 MPI\_Init(&argc, &argv);  
 MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);  
 MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);  
 send\_to = (rank + 1) % size;  
 recv\_from = (rank - 1 + size) % size;  
  
 message = rank;  
 int received\_message;  
 MPI\_Sendrecv(&message, 1, MPI\_INT, send\_to, 0,  
 &received\_message, 1, MPI\_INT, recv\_from, 0,  
 MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);  
 printf("Process %d received message %d from process %d\n", rank, received\_message, recv\_from);  
 MPI\_Finalize();  
 return 0;  
}

Output:

A screenshot of a computer program

AI-generated content may be incorrect.

**Problem 3:**

import threading

import queue

import time

import random

NUM\_PROCESSES = 3

printer\_lock = threading.Lock()

class Process(threading.Thread):

def \_\_init\_\_(self, pid, token\_queues):

super().\_\_init\_\_()

self.pid = pid

self.token\_queues = token\_queues

self.request\_print = False

def request\_to\_print(self):

self.request\_print = True

def run(self):

while True:

token = self.token\_queues[self.pid].get() # Wait for token

print(f"P{self.pid+1} received token.")

if self.request\_print:

with printer\_lock:

print(f"\*\*\* P{self.pid+1} is printing \*\*\*")

time.sleep(1) # Simulate printing

print(f"\*\*\* P{self.pid+1} finished printing \*\*\*")

self.request\_print = False

# Pass the token to the next process in ring

next\_pid = (self.pid + 1) % NUM\_PROCESSES

print(f"P{self.pid+1} passing token to P{next\_pid+1}")

time.sleep(1)

self.token\_queues[next\_pid].put("TOKEN")

# Create token queues for each process

token\_queues = [queue.Queue() for \_ in range(NUM\_PROCESSES)]

# Create and start the processes

processes = [Process(pid=i, token\_queues=token\_queues) for i in range(NUM\_PROCESSES)]

for p in processes:

p.start()

# Initially give token to P1

token\_queues[0].put("TOKEN")

# Simulate print requests

def generate\_requests():

while True:

time.sleep(random.randint(2, 4))

pid = random.randint(0, NUM\_PROCESSES - 1)

processes[pid].request\_to\_print()

print(f"P{pid+1} has requested to print.")

# Run request generator in a separate thread

request\_thread = threading.Thread(target=generate\_requests)

request\_thread.daemon = True

request\_thread.start()

Output:

