Assignment No. - 06

Aim:- Implementing Bully & Ring Algorithm for Leader Election.

Tools and environment:- C++ programming environment.

Theory :-

Election algorithms choose a process from a group of processors to act as a coordinator. If coordinator process crashes due to some reasons, then a new coordinator is elected on other processor. Election algorithm basically determines where a new copy of coordinator should be restarted. Election algorithm assumes that every active process in system has a unique priority number. The process with highest priority will be chosen as a new coordinator. Hence, when a coordinator fails, this algorithm elects that active process which has highest priority number. Then this number is send to every active process in distributed system. We have two election algorithms for two different configurations of a distributed systems.

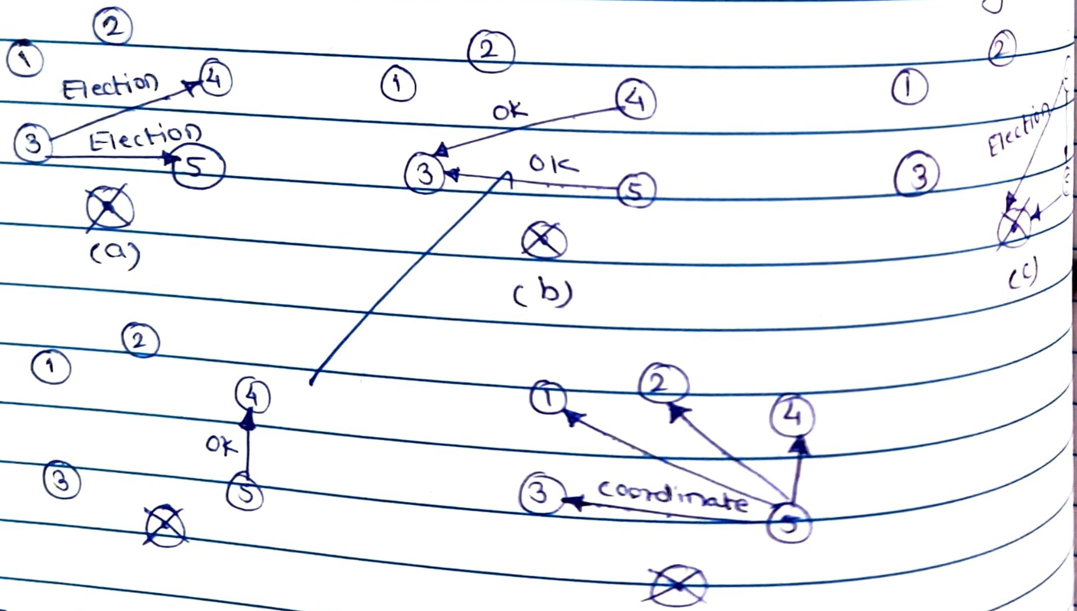
- 1) **The Bully Algorithm:-**

This algorithm applies to system where every process can send a message to every other process in the system.

Algorithm:-

Suppose process P sends a message to coordinator. If the coordinator does not respond to it within a time interval T, then it is assumed tha has failed. Now process P sends an elec to every process with high priority nu for responses, if no one responds for then process P elects itself as a coordina sends a message to all lower priority n processes that it is elected as their ne However, if an answer is received within any other process Q, (I) Process P again w time interval T' to receive another messa that it has been elected as coordinator.

(II) If Q doesn't responds within time inte it is assumed to have failed and algorithm



2) The Ring Algorithm :-

This algorithm applies to systems organi ring (logically or physically). In this algorith that the link betn the process are unidirection every process can message to the process on only. Data structure that this algorithm us list, a list that has a priority no. of all ac the system.

Algorithm :-

If process P1 detects a coordinator failure, it creates new active list which is empty initially. It sends election message to its neighbour on right & adds number 1 to its active list.

If process P2 receives message elect from processes on left, it responds in 3 ways:-

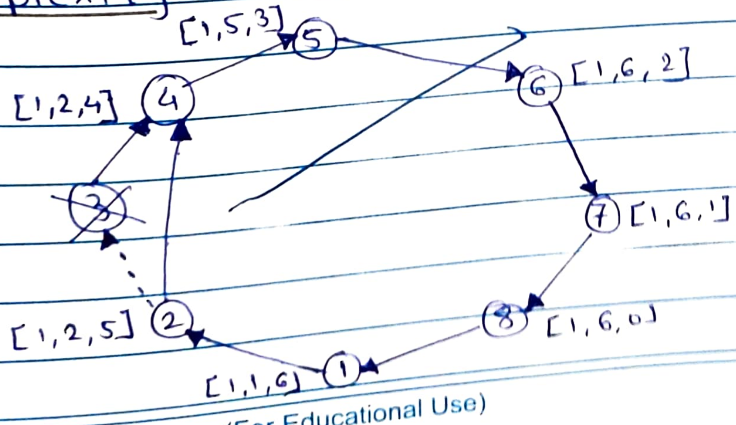
i) If message received does not contain 1 in active list then P1 adds 2 to its active list & forwards the message.

ii) If this is the first election message it has received or sent, P1 creates new active list with numbers 1 and 2. It then sends election message 1 followed by 2.

iii) If process P1 receives its own election message 1 then active list for P1 now contains numbers of all the active processes in the system. Now Process P1 detects highest priority number from list and elects it as the new coordinator.

Time Complexity :- O(n^2) in the worst case scenario, where n is the number of processes.

Space complexity :- O(n)



- Conclusion:-

The bully algorithm is a type of Electio
which is mainly used for choosing a coor
Hence in a distributed system, we need su
algorithms such as bully and ring to get
coordinator that performs functions neede
processes.

NAME: PALLAVI K. CHOPADE
ROLL NO.: 14
PRN NO.: 72036169K
BE (IT)
LP - V

```
#Bully.py

from statistics import mode

class Process:
    def __init__(self, process_id, total_count):
        self.process_id = process_id
        self.total_count = total_count
        self.leader_id = -1
        self.is_active = True

    def crash(self):
        self.is_active = False

    def start(self):
        self.is_active = True

    def is_leader(self):
        if self.process_id == self.leader_id:
            return True
        return False

    def set_leader(self, leader):
        self.leader_id = leader

    def get_leader(self):
        return self.leader_id

    def sendRequest(self, toProcess):
        print(f"Sending request to process {toProcess.process_id} from
        {self.process_id}")
        if(toProcess.reciveRequest(self.process_id)):
            print(f"Ok recived from {toProcess.process_id}")
            self.set_leader(toProcess.process_id)
        else:
            print(f"No response from {toProcess.process_id}")


    def reciveRequest(self, fromProcess):
        if(self.is_active):
            print(f"Recived request from process {fromProcess}.")
            return self.recivedMessage()
        return False

    def recivedMessage(self):
            return True;

class Bully:
    def __init__(self, total_count):
        self.processes = []
        self.total_count = total_count
        # self.leader = None

    def intiailzeProcesses(self):
        self.processes = []
        for i in range(self.total_count):
            self.processes.append(Process(i, total_count = self.total_count))
        self.elect_leader()
        self.coordinator()

    def elect_leader(self, current=0):
        for i in range(current, self.total_count):
            if self.processes[i].is_active:
```

```
            # [self.processes[i].sendRequest(self.processes[j]) for j in range(i,
            self.total_count)]
            for j in range(i+1, self.total_count):
                if(self.processes[j].is_active):
                    self.processes[i].sendRequest(self.processes[j])
                elif(not self.processes[j].is_active and i+1==self.total_count-1):
                    self.processes[i].sendRequest(self.processes[i])

            if self.processes[i].get_leader()==-1:
                self.processes[i].sendRequest(self.processes[i])
            # if(i==self.total_count-1):
            # self.processes[i].sendRequest(self.processes[i])

    def crash(self, crash_id):
        if(crash_id<self.total_count and crash_id>=0):
            self.processes[crash_id].crash()
            # print(f"Process id {Process.process_id} crashed.")
            if(self.processes[crash_id].is_leader()):
                print("Leader process Down.\n Initaling the leader lookout.")
                self.elect_leader(0)


    def start(self, process_id):
        if(self.processes[process_id].is_active):
            print("Process already active")
        else:
            self.processes[process_id].start()
            self.elect_leader()
            # if(self.processes[process_id].is_active):
            #   if process_id>self.processes[self.leader].get_leader():
                    # self.elect_leader(self.leader)


    def coordinator(self):
        leader = []
        for p in self.processes:

            if p.is_active:
                print(p.get_leader())
                leader.append(p.get_leader())

        self.leader = mode(leader)
#Driver.py

from Bully import Bully
#Dummy Processes

process_count = int(input("Enter Number of Processes:"))
bully = Bully(process_count)
bully.intiailzeProcesses()

state = True

while state:
    print("1. Initalize the process\n2. Bring Down process\n3. Activate Process\n4. Exit
    \n5. Current Coordinator\n")
    choice = int(input())
    if(choice==1):
        bully.intiailzeProcesses()

    elif(choice==2):
        crash_id = int(input("Enter the process you want to crash"))
        bully.crash(crash_id)

    elif(choice==3):
        process_id = int(input("Enter the process you want to start"))
        bully.start(process_id)

    elif(choice==4):
        state=False
        print("Exiting the program")

    elif(choice==5):
        bully.coordinator()
    else:
        print("Invalid Input")


#Ring.py

class Pro:
    def __init__(self, id):
        self.id = id
        self.act = True


class GFG:
    def __init__(self):
        self.TotalProcess = 0
        self.process = []

    def initialiseGFG(self):
        print("No of processes 5")
        self.TotalProcess = 5
        self.process = [Pro(i) for i in range(self.TotalProcess)]

    def Election(self):
        print("Process no " + str(self.process[self.FetchMaximum()].id) + " fails")
        self.process[self.FetchMaximum()].act = False
        print("Election Initiated by 2")
        initializedProcess = 2

        old = initializedProcess
        newer = old + 1

        while (True):
            if (self.process[newer].act):                 str(self.process[old].id) + " pass Election(" +
                print("Process " + str(self.process[old].id) + ") to" + str(self.process[newer].id))
                str(self.process[old].id) + ") to" + str(self.process[newer].id))
                old = newer
            newer = (newer + 1) % self.TotalProcess
            if (newer == initializedProcess):
                break
                                                       str(self.process[self.FetchMaximum()].id) + " becomes
        print("Process " + str(self.process[self.FetchMaximum()].id) + " becomes
        coordinator")
        coord = self.process[sclf.FetchMaximum()].id

        old = coord
        newer = (old + 1) % self.TotalProcess
        while (True):
            if (self.process[newer].act):
                print("Process " + str(self.process[old].id) + " pass Coordinator(" +
                str(coord) + ") message to process " + str(self.process[newer].id))
                old = newer
            newer = (newer + 1) % self.TotalProcess
            if (newer == coord):
                print("End Of Election ")
                break

    def FetchMaximum(self):
        maxId = -9999
        ind = 0
        for i in range(self.TotalProcess):
            if (self.process[i].act and self.process[i].id > maxId):
                maxId = self.process[i].id
                ind = i
        return ind
def main():
    object = GFG()
    object.initialiseGFG()
    object.Election()

if __name__ == "__main__":
    main()
```

✓

Assignment No.- 06

Name: -Pallavi Kamlakar Chopade.

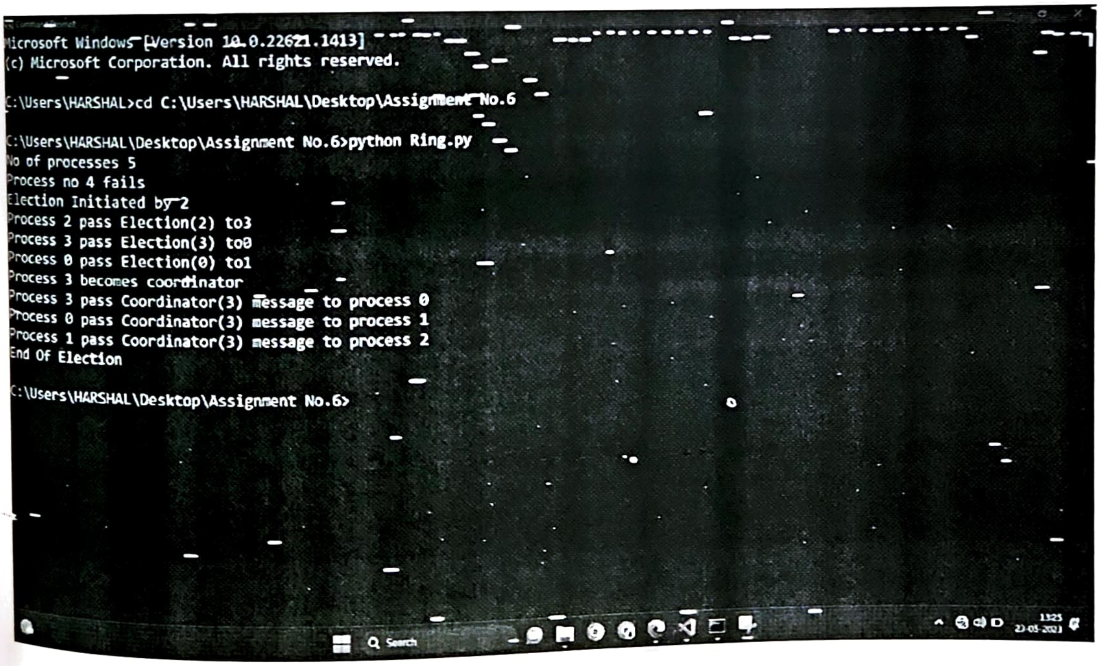
Roll No.: - 14

PRN No.: - 72036169K

Subject: - Distributed Systems

Class: - BE(IT)

Ring Algorithm Output: -

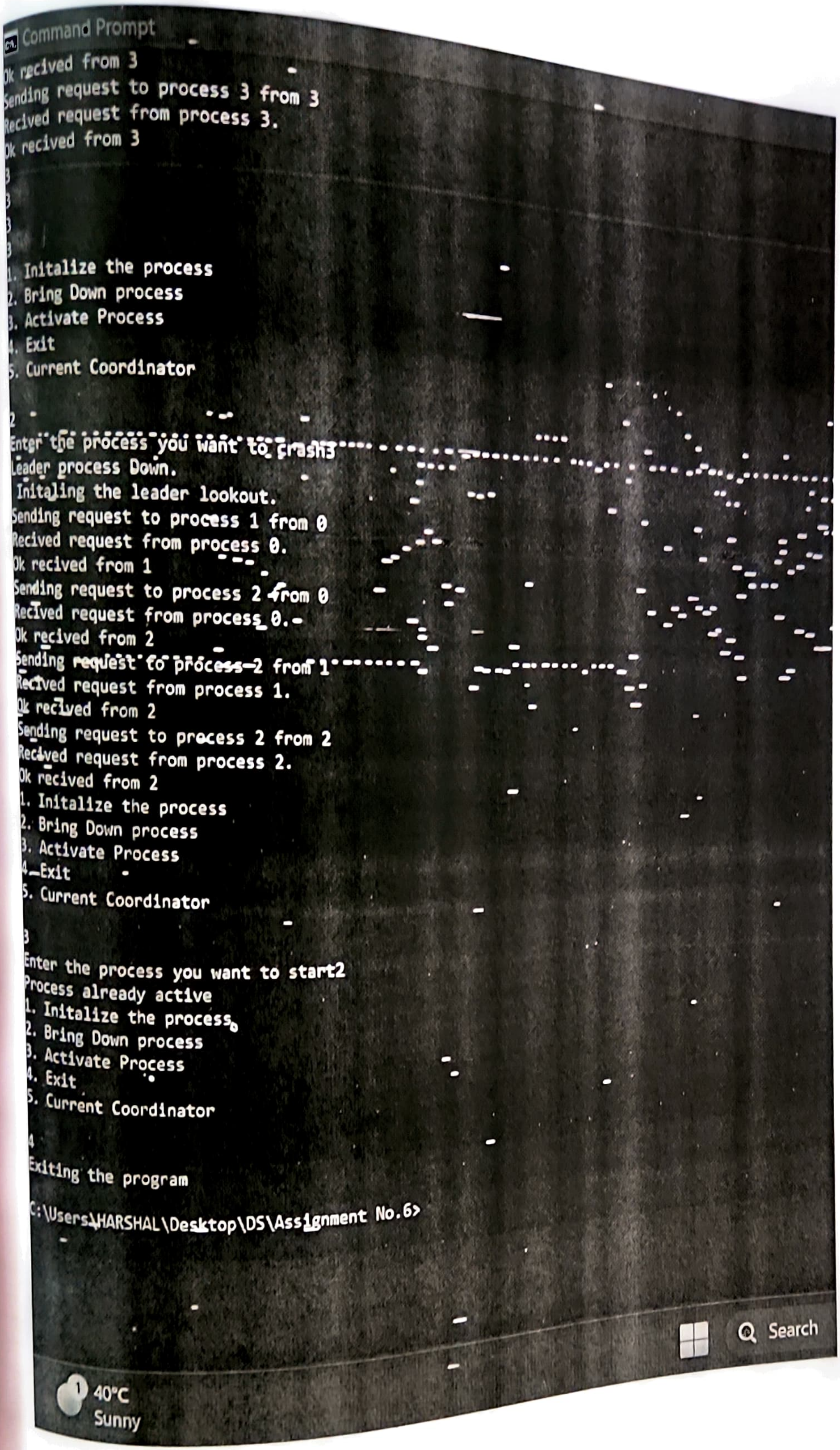


```
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HARSHAL>cd C:\Users\HARSHAL\Desktop\Assignment No.6

C:\Users\HARSHAL\Desktop\Assignment No.6>python Ring.py
No of processes 5
Process no 4 fails
Election Initiated by 2
Process 2 pass Election(2) to3
Process 3 pass Election(3) to0
Process 0 pass Election(0) to1
Process 3 becomes coordinator
Process 3 pass Coordinator(3) message to process 0
Process 0 pass Coordinator(3) message to process 1
Process 1 pass Coordinator(3) message to process 2
End Of Election

C:\Users\HARSHAL\Desktop\Assignment No.6>
```



```
Ok recived from 3
Sending request to process 3 from 3
Recived request from process 3.
Ok recived from 3
3
3
3
3
1. Initalize the process
2. Bring Down process
3. Activate Process
4. Exit
5. Current Coordinator

2
Enter the process you want to crash3
Leader process Down.
 Initaling the leader lookout.
Sending request to process 1 from 0
Recived request from process 0.
Ok recived from 1
Sending request to process 2 from 0
Recived request from process 0.
Ok recived from 2
Sending request to process 2 from 1
Recived request from process 1.
Ok recived from 2
Sending request to process 2 from 2
Recived request from process 2.
Ok recived from 2
1. Initalize the process
2. Bring Down process
3. Activate Process
4. Exit
5. Current Coordinator

3
Enter the process you want to start2
Process already active
1. Initalize the process
2. Bring Down process
3. Activate Process
4. Exit
5. Current Coordinator

4
Exiting the program

C:\Users\HARSHAL\Desktop\DS\Assignment No.6>
```