Assignment No. 6

Name: Viraj Sarjerao Kawade

Roll No: 043

BE-(IT)

Subject: LP-V (Distributed System)

Code: Bully Algorithm

```
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)
```

```
23.
24.     elif(choice==4):
25.         state=False
26.         print("Exiting the program")
27.
28.     elif(choice==5):
29.         bully.coordinator()
30.     else:
31.         print("Invalid Input")
32.
```

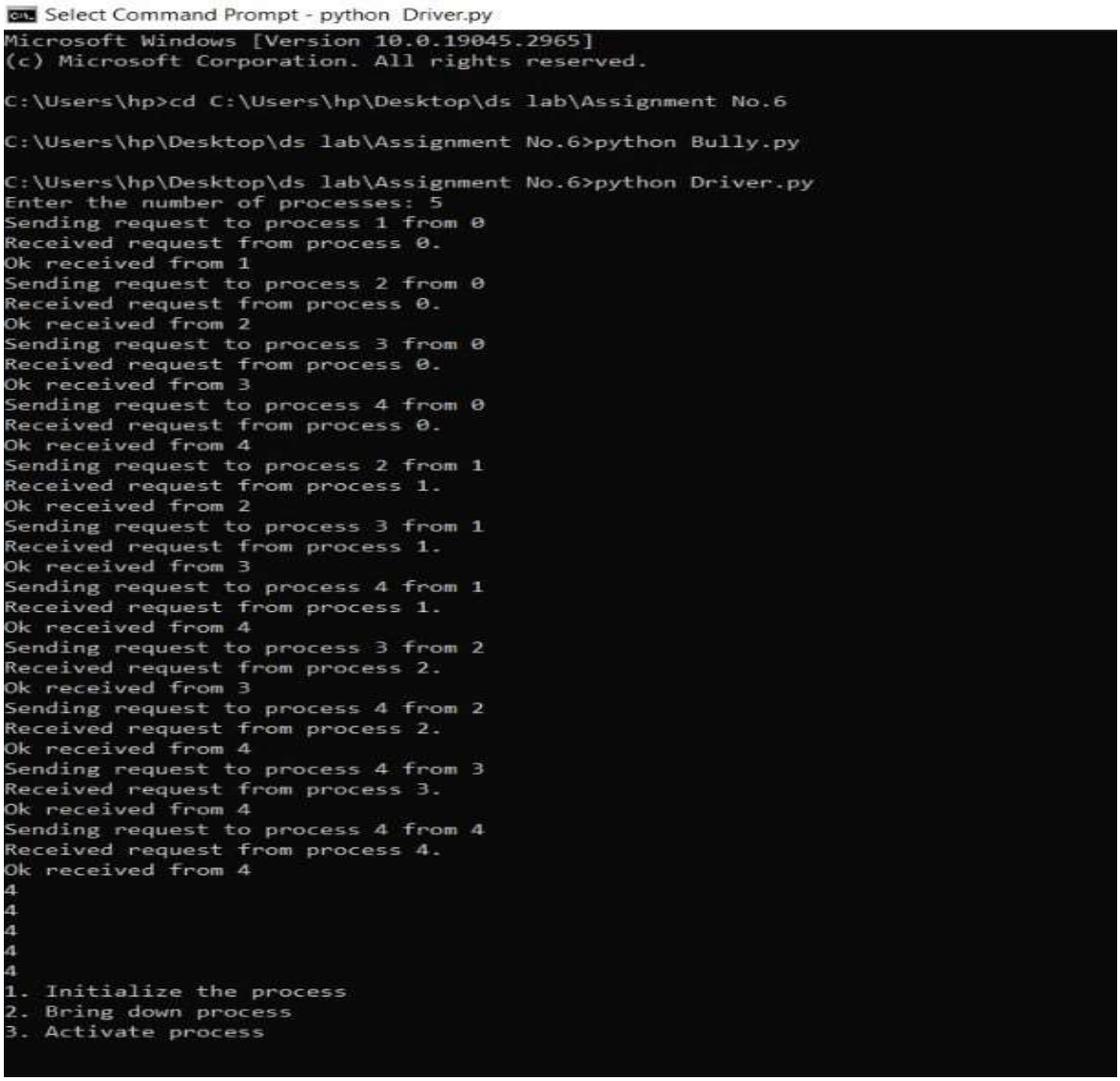
Ring Algorithm

```
 1. class Pro:
 2.       def__init__(self, id):
 3.                self.id = id
 4.                self.act = True
 5.
 6. class GFG:
 7.       def__init__(self):
 8.                self.TotalProcess = 0
 9.                self.process = []
10.
11.       def initialiseGFG(self):
12.                print("No of processes 5")
13.                self.TotalProcess = 5
14.                self.process = [Pro(i) for i in range(self.TotalProcess)]
15.
16.       def Election(self):
17.                print("Process no " + str(self.process[self.FetchMaximum()].id) + " fails")
18.                self.process[self.FetchMaximum()].act = False
19.                print("Election Initiated by 2")
20.                initializedProcess = 2
21.
22.                old = initializedProcess
23.                newer = old + 1
24.
25.                while (True):
26.                          if (self.process[newer].act):
27.                                    print("Process " + str(self.process[old].id) + " pass
Election(" + str(self.process[old].id) + ") to" + str(self.process[newer].id))
28.                                    old = newer
29.                          newer = (newer + 1) % self.TotalProcess
30.                          if (newer == initializedProcess):
31.                                    break
32.32.
33.                print("Process " + str(self.process[self.FetchMaximum()].id) + " becomes
coordinator")
34.                coord = self.process[self.FetchMaximum()].id
35.
36.                old = coord
37.                newer = (old + 1) % self.TotalProcess
38.                while (True):
39.                          if (self.process[newer].act):
40.                                    print("Process " + str(self.process[old].id) + " pass
Coordinator(" + str(coord) + ") message to process " + str(self.process[newer].id))
41.                                    old = newer
42.                          newer = (newer + 1) % self.TotalProcess
43.                          if (newer == coord):
44.                                    print("End Of Election ")
45.                                    break
46.46.
47.       def FetchMaximum(self):
48.                maxId = -9999
49.                ind = 0
50.                for i in range(self.TotalProcess):
51.                          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()

```

Output:

After Implementing The Bully Algorithm:



After Implementing The Ring Algorithm: