**Institute of Computer Technology**

**B. Tech Computer Science and Engineering**

**Sub: Operating Systems**

**Banker’s Algorithm**

**CODE:**

import numpy as np

total\_available = []

data = []

current\_available = []

safe\_order = []

def readData():

with open('input\_for\_bankers\_algo') as file:

global total\_available

total\_available = np.array(eval(file.readline().split('=')[1].replace('\n','')))

file.readline()

for line in file:

rdata = line.replace('\n','').split()

data.append(

dict(PID=rdata[0], AVAIL=np.array(eval(rdata[1])), MAX=np.array(eval(rdata[2])),DONE=False)

)

def calculate\_resources\_needed():

avails = np.array([d['AVAIL'] for d in data])

global current\_available

current\_available = total\_available - np.sum(avails,axis=0)

for d in data: d['NEED'] = d['MAX'] - d['AVAIL']

def process\_order\_for\_safe\_state():

global current\_available, safe\_order

while not all([d['DONE'] for d in data]):

at\_least\_one\_is\_allocated = False

for d in data:

if not d['DONE']:

if all(current\_available - d['NEED'] >=0 ):

at\_least\_one\_is\_allocated = True

safe\_order.append(d['PID'])

d['DONE'] = True

current\_available+=d['AVAIL']

if not at\_least\_one\_is\_allocated:

print('Resource Allocation not Safe..')

exit(-1)

def display():

print('PID\t',' ALLOCATION\t',' MAX\t\t',' NEED')

for d in data:

print('{PID}\t\t{AVAIL}\t\t{MAX}\t\t{NEED}'.format(\*\*d))

print('\nSafe Order : ', ' -> '.join(safe\_order))

readData()

calculate\_resources\_needed()

process\_order\_for\_safe\_state()

display()

**OUTPUT:**

