# Practice Part 12

6.1

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Curre | nt Allo | cation |  | Max | dema | nd |  | Need | s |  |  |
| process | r1 | r2 | r3 | r4 | r1 | r2 | r3 | r4 | r1 | r2 | r3 | r4 |
| p1 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 | 0 | 0 | 0 | 0 |
| p2 | 1 | 0 | 0 | 1 | 1 | 5 | 1 | 1 | 0 | 5 | 1 | 0 |
| p3 | 1 | 3 | 0 | 4 | 2 | 3 | 5 | 6 | 1 | 0 | 5 | 2 |
| p4 | 0 | 6 | 3 | 2 | 0 | 6 | 7 | 2 | 0 | 0 | 4 | 0 |
| p5 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 | 0 | 6 | 4 | 2 |
| Total | 2 | 9 | 5 | 13 |  |  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| r1 | r2 | r3 | r4 |
| 1 | 5 | 2 | 0 |

Available/Resources: <1, 5, 2, 0>

1. P1<0, 0, 0, 0> <= <1, 5, 2, 0> -> True

New Available = Allocation + Available

= <0, 0, 1, 2> + <1, 5, 2, 0>

= <1, 5, 3, 2>

2) P2<0, 5, 1, 0> <= <1, 5, 3, 2> -> True

= <1, 0, 0, 1> + <1, 5, 3, 2>

= <2, 5, 3, 3>

3) P3<1, 0, 5, 2> <= <2, 5, 3, 3> ->False

4) P4<0, 0, 4, 0> <= <2, 5, 3, 3> ->False

5) P5<0, 6, 4, 2> <= <2, 5, 3, 3> ->False

Process in unsafe state

6.3

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Allocation |  |  |  | Max |  |  |  | Need |  |  |  |
|  | A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 2 | 0 | 2 | 1 | 9 | 5 | 5 | 5 | 7 | 5 | 3 | 4 |
| P1 | 0 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 2 | 1 | 2 | 2 |
| P2 | 4 | 1 | 0 | 2 | 7 | 5 | 4 | 4 | 3 | 4 | 4 | 2 |
| P3 | 1 | 0 | 0 | 1 | 3 | 3 | 3 | 2 | 2 | 3 | 3 | 1 |
| P4 | 1 | 1 | 0 | 0 | 5 | 2 | 2 | 1 | 4 | 1 | 2 | 1 |
| P5 | 1 | 0 | 1 | 1 | 4 | 4 | 4 | 4 | 3 | 4 | 3 | 3 |
| Total | 9 | 3 | 4 | 6 |  |  |  |  |  |  |  |  |

Available

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | D |
| 6 | 3 | 5 | 4 |

Resources type: A(15 instances), B(6 instances), C(9 instances), D(10 instances)

1. Verify that the Available array has been calculated correctly.

Available = Instances - Total Allocation

6 = 15 - 9 True

3 = 6 - 3 True

5 = 9 - 4 True

4 = 10 - 6 True

b. Calculate the Need matrix.

Calculated

c. Show that the current state is safe, that is, show a safe sequence of processes. In addition, to the sequence show how the Available (working array) changes as each process terminates.

P0<7, 5, 3, 4> <= <6, 3, 5, 4> -> False

P1<2, 1, 2, 2> <= <6, 3, 5, 4> -> True

New Available = Allocation + Available

= <0, 1, 1, 1> + <6, 3, 5, 4>

= <6, 4, 6, 5>

P2<3, 4, 4, 2> <= <6, 4, 6, 5> -> True

= <4, 1, 0, 2> + <6, 4, 6, 5>

= <10, 5, 6, 7>

P3<2, 3, 3, 1> <= <10, 5, 6, 7> -> True

= <1, 0, 0, 1> + <10, 5, 6, 7>

= <11, 5, 6, 8>

P4<4, 1, 2, 1> <= <11, 5, 6, 8> -> True

= <1, 1, 0, 0> + <11, 5, 6, 8>

= <12, 6, 6, 8>

P5<3, 4, 3, 3> <= <12, 6, 6, 8> -> True

= <1, 0, 1, 1> + <12, 6, 6, 8>

= <13, 6, 7, 9>

P3<2, 3, 3, 1> <= <13, 6, 7, 9> -> True

= <1, 0, 0, 1> + <13, 6, 7, 9>

= <14, 6, 7, 10>

Process in a safe state

d. Given the request (3,2,3,3) from Process P5. Should this request be granted? Why or why not?

1) Request5 <= Need5

<3, 2, 3, 3> <= <3, 4, 3, 3>

Satisfied

2) Request5 <= Available

<3, 2, 3, 3> <= <6, 3, 5, 4>

Satisfied

3) Available = Available - Request5 = <6, 3, 5, 4> - <3, 2, 3, 3> = (3, 1, 2, 1)

Allocation5 = Allocation5 + Request5 = <1, 0, 1, 1> + <3, 2, 3, 3> = (4, 2, 4, 4)

Need5 = Need5 - Request5 = <3, 4, 3, 3> - <3, 2, 3, 3> = (0, 2, 0, 0)

4) Work = Available = (3, 1, 2, 1)

P0<7, 5, 3, 4> <= <3, 1, 2, 1> -> False

P1<2, 1, 2, 2> <= <3, 1, 2, 1> -> False

P2<3, 4, 4, 2> <= <3, 1, 2, 1> -> False

P3<2, 3, 3, 1> <= <3, 1, 2, 1> -> False

P4<4, 1, 2, 1> <= <3, 1, 2, 1> -> False

P5<3, 4, 3, 3> <= <3, 1, 2, 1> -> False

Since process is not in safe state, it means that Process 5 shouldn’t be granted

## Laboratory Part

def main():

processes = int(input("Amount of processes : "))

resources = int(input("Amount of resources : "))

print("\n Allocation of process: ")

currently\_allocated = [[int(i) for i in input(f"process {j + 1} : ").split()] for j in range(processes)]

print("\n Resources of process: ")

max\_need = [[int(i) for i in input(f"process {j + 1} : ").split()] for j in range(processes)]

allocated = [0] \* resources

for i in range(processes):

for j in range(resources):

allocated[j] += currently\_allocated[i][j]

available = []

for i in range(4):

available.append(int(input("\n Available: ")))

print(f"total available resources : {available}\n")

running = [True] \* processes

count = processes

while count != 0:

safe = False

for i in range(processes):

if running[i]:

executing = True

for j in range(resources):

if max\_need[i][j] - currently\_allocated[i][j] > available[j]:

executing = False

break

if executing:

print(f"process {i + 1} is executing")

running[i] = False

count -= 1

safe = True

for j in range(resources):

available[j] += currently\_allocated[i][j]

break

if not safe:

print("the processes are in an unsafe state.")

break

print(f"the process is in a safe state.\navailable resources : {available}\n")

if \_\_name\_\_ == '\_\_main\_\_':

main()

