**Student Name:** Y.mohana sri vamsy

**Student ID:** 11800983

**Email Address:** consumables.yandamuri@gmail.com

**GitHub Link:** <https://github.com/vamsy5>

**Question:22** Consider following and Generate a solution in C to find whether the system

is in safe state or not?

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Available | | | | Processes | Allocation | | | | Max | | | |
| A | B | C | D | A | B | C | D | A | B | C | D |
| 1 | 5 | 2 | 0 | P0 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 |
|  | | | | P1 | 1 | 0 | 0 | 0 | 1 | 7 | 5 | 0 |
| P2 | 1 | 3 | 5 | 4 | 2 | 3 | 5 | 6 |
| P3 | 0 | 6 | 3 | 2 | 0 | 6 | 5 | 2 |
| P4 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 |

**Description:-**

The banker’s algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an “s-state” check to test for possible activities, before deciding whether allocation should be allowed to continue.

Let **‘n’**be the number of processes in the system and **‘m’**be the number of resources types.

**Available :**

* It is a 1-d array of size **‘m’** indicating the number of available resources of each type.
* Available[ i ] = k means there are **‘k’** instances of resource type **Rj**

**Max :**

* It is a 2-d array of size ‘**n\*m’**that defines the maximum demand of each process in a system.
* Max[ i, j ] = k means process **Pi** may request at most **‘k’** instances of resource type **Rj.**

**Allocation :**

* It is a 2-d array of size**‘n\*m’**that defines the number of resources of each type currently allocated to each process.
* Allocation[ i, j ] = k means process **Pi** is currently allocated **‘k’** instances of resource type **Rj**

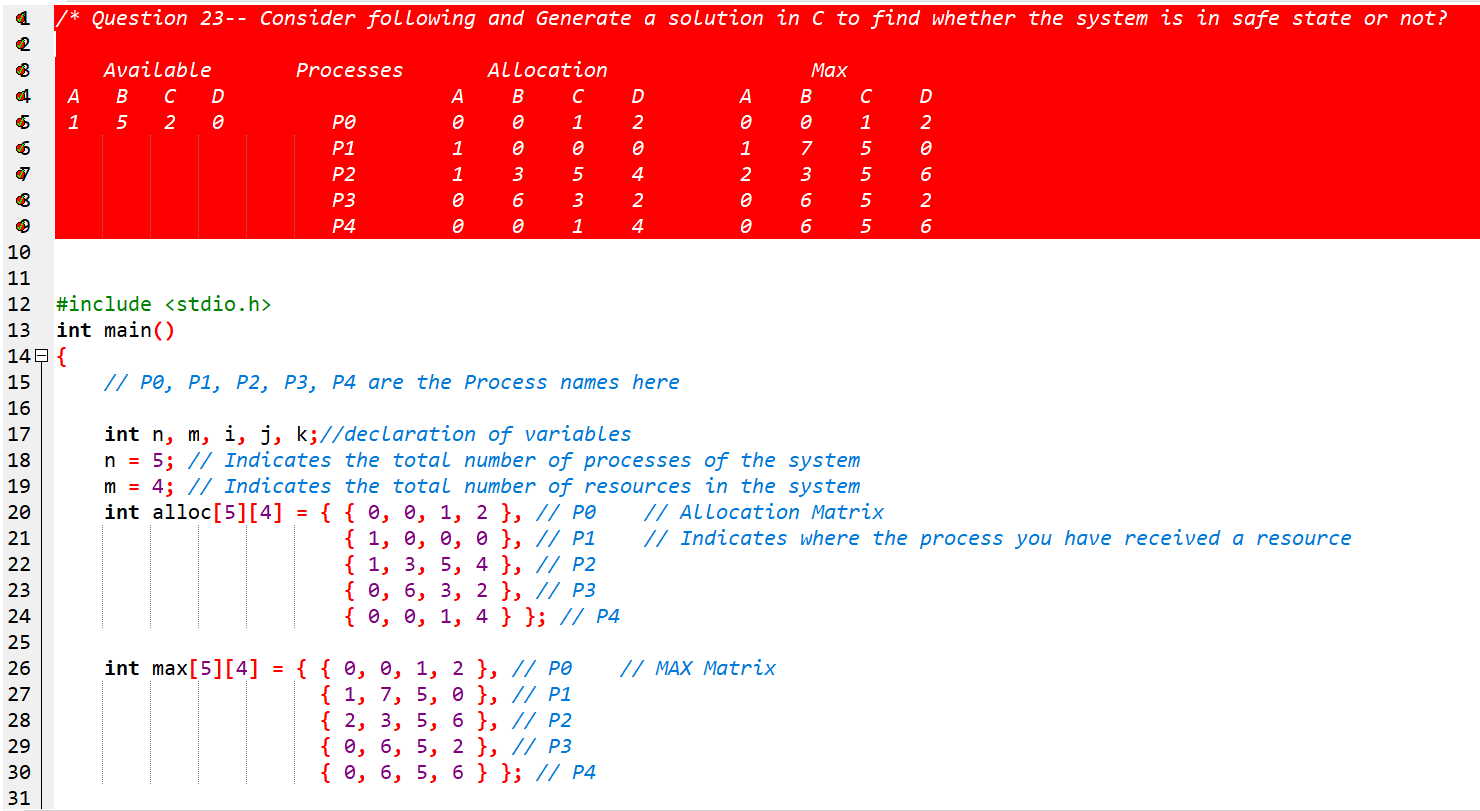
**Need :**

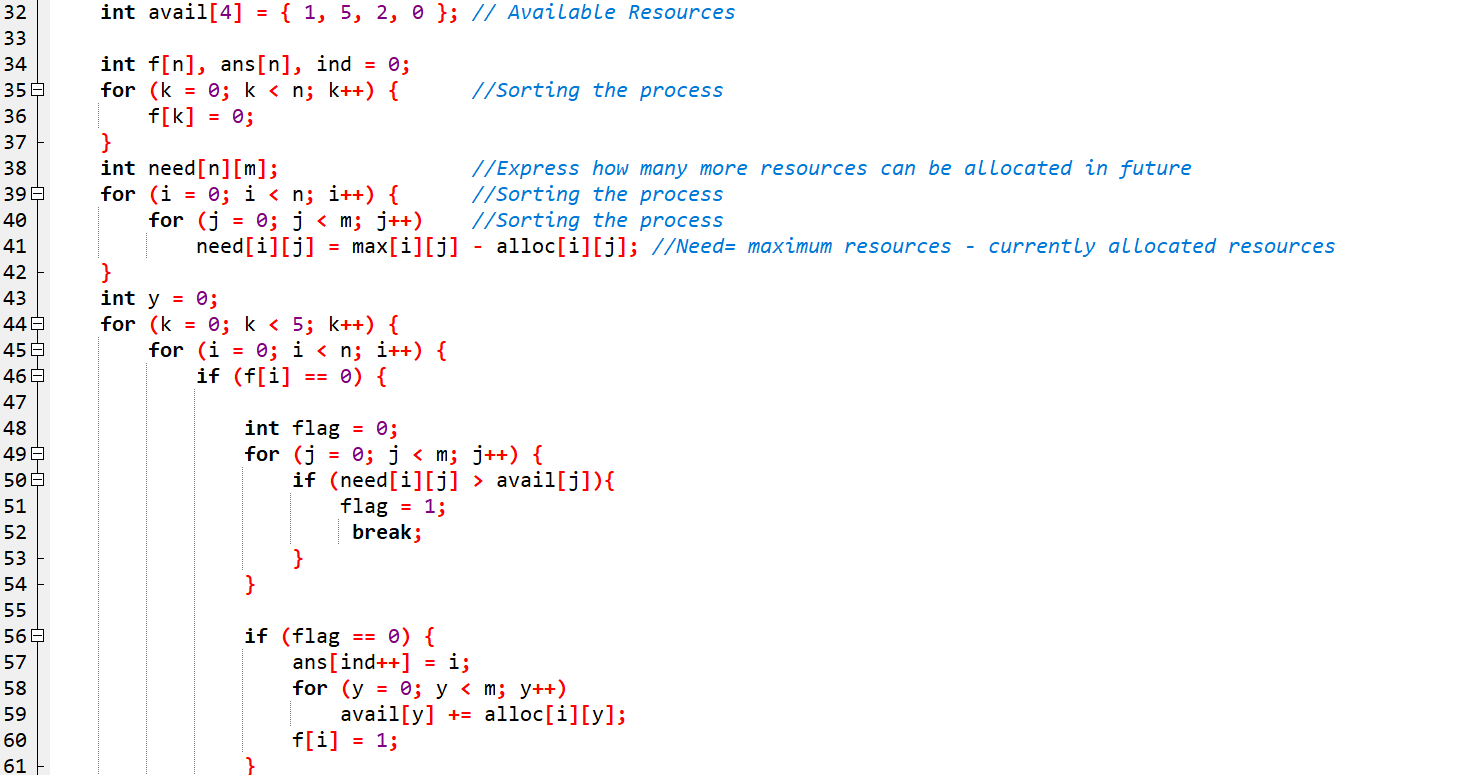
* It is a 2-d array of size **‘n\*m’** that indicates the remaining resource need of each process.
* Need [ i,   j ] = k means process **Pi** currently need **‘k’** instances of resource type **Rj**

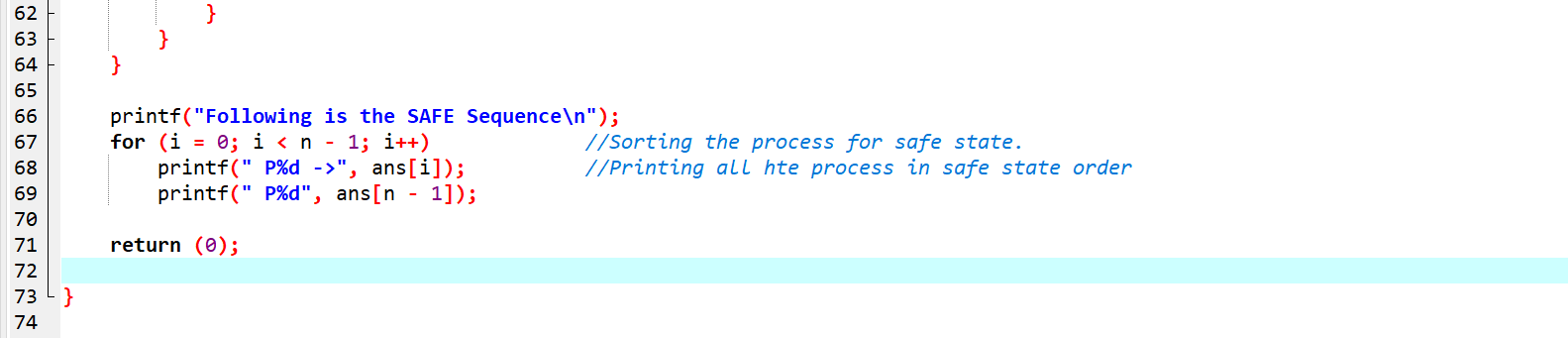
for its execution.

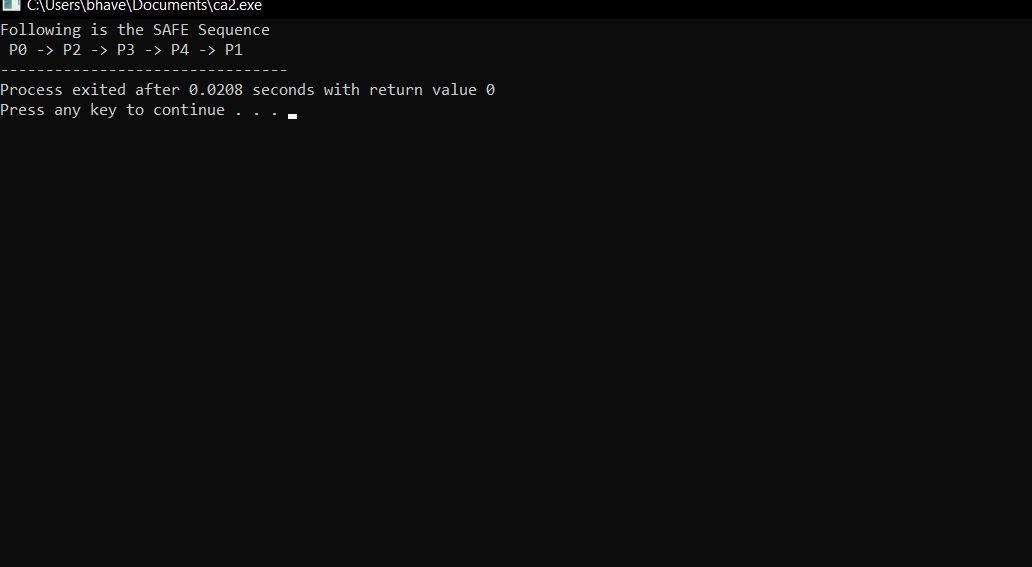
* Need [ i,   j ] = Max [ i,   j ] – Allocation [ i,   j]

**Code:-(Predefined values in the code)**

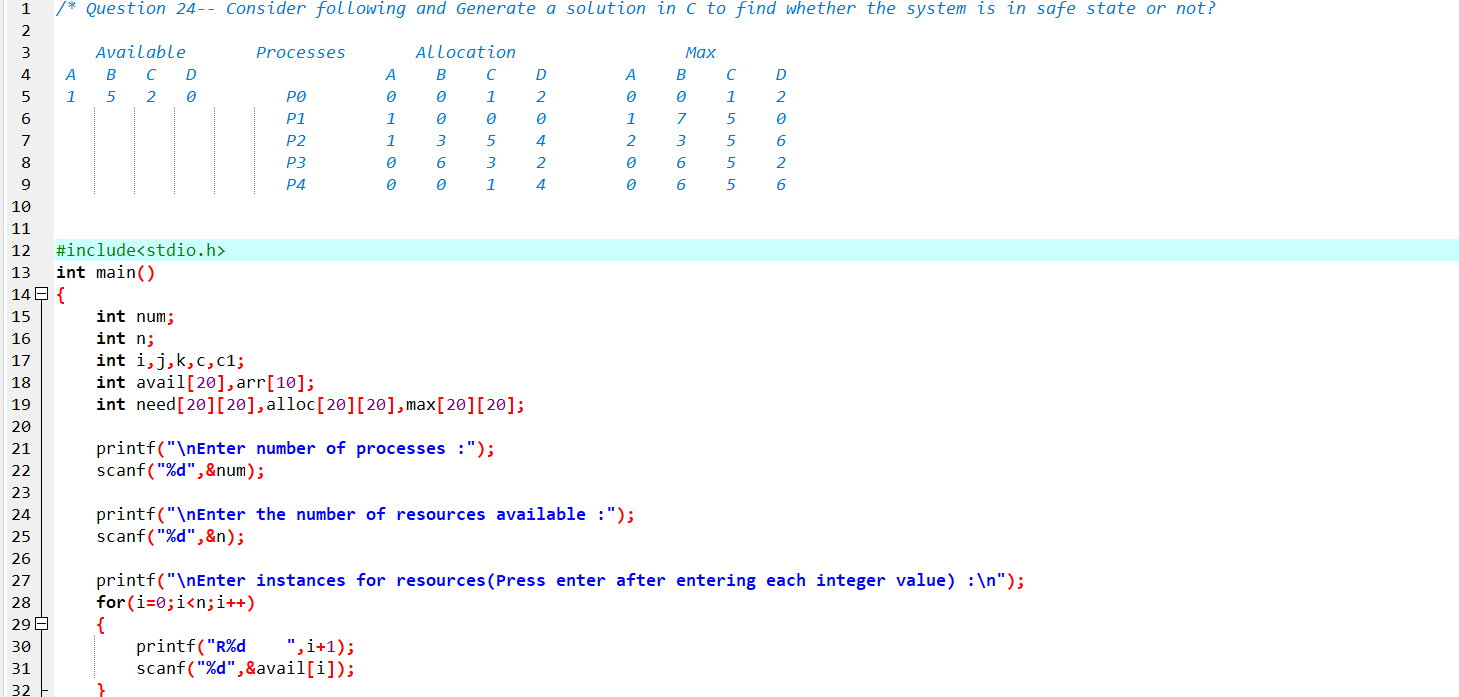
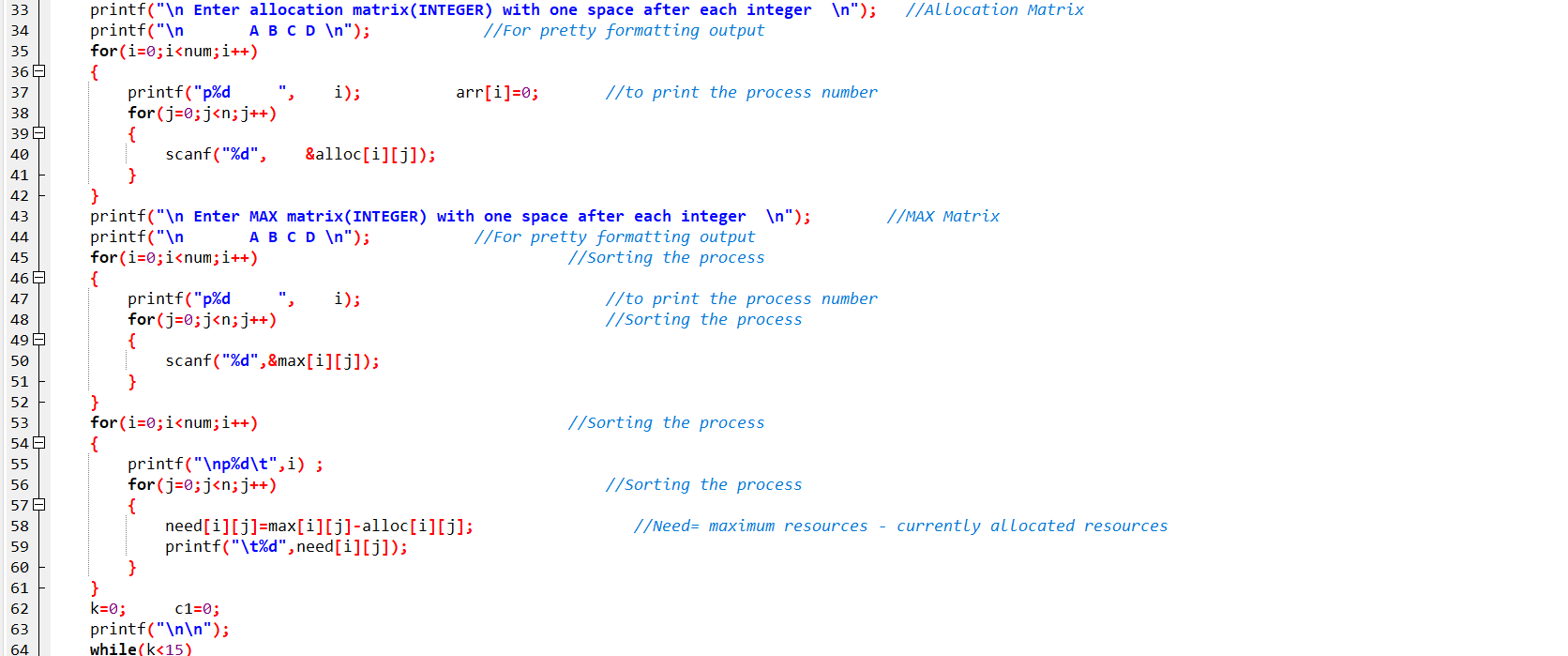
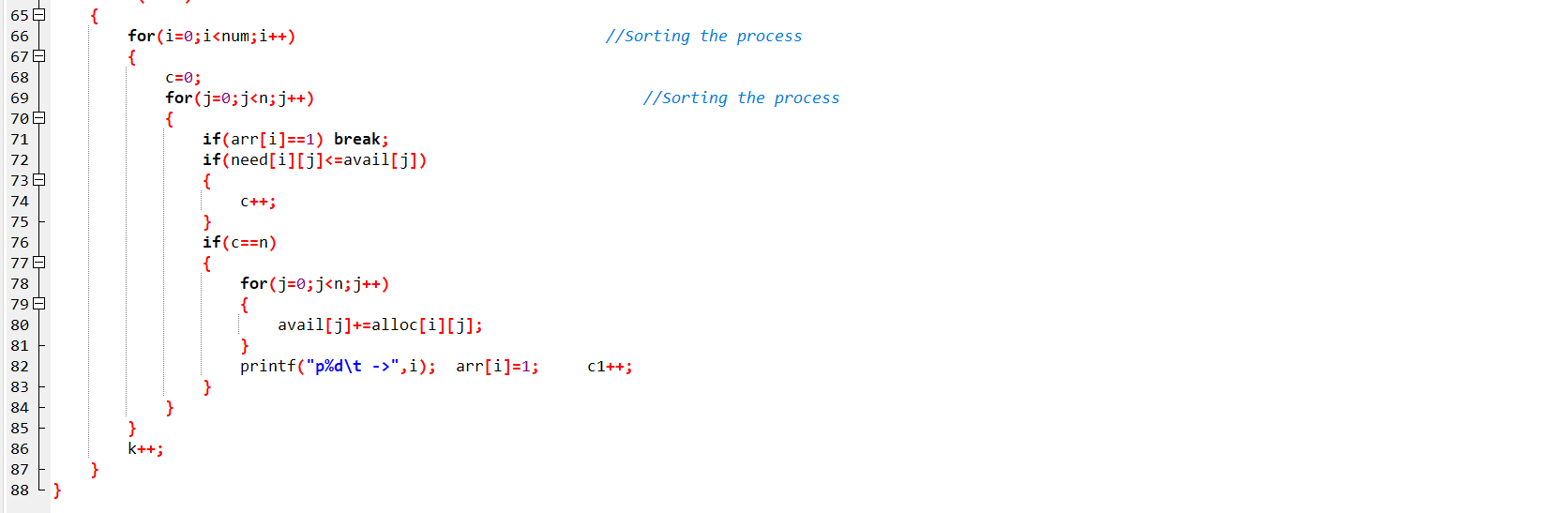


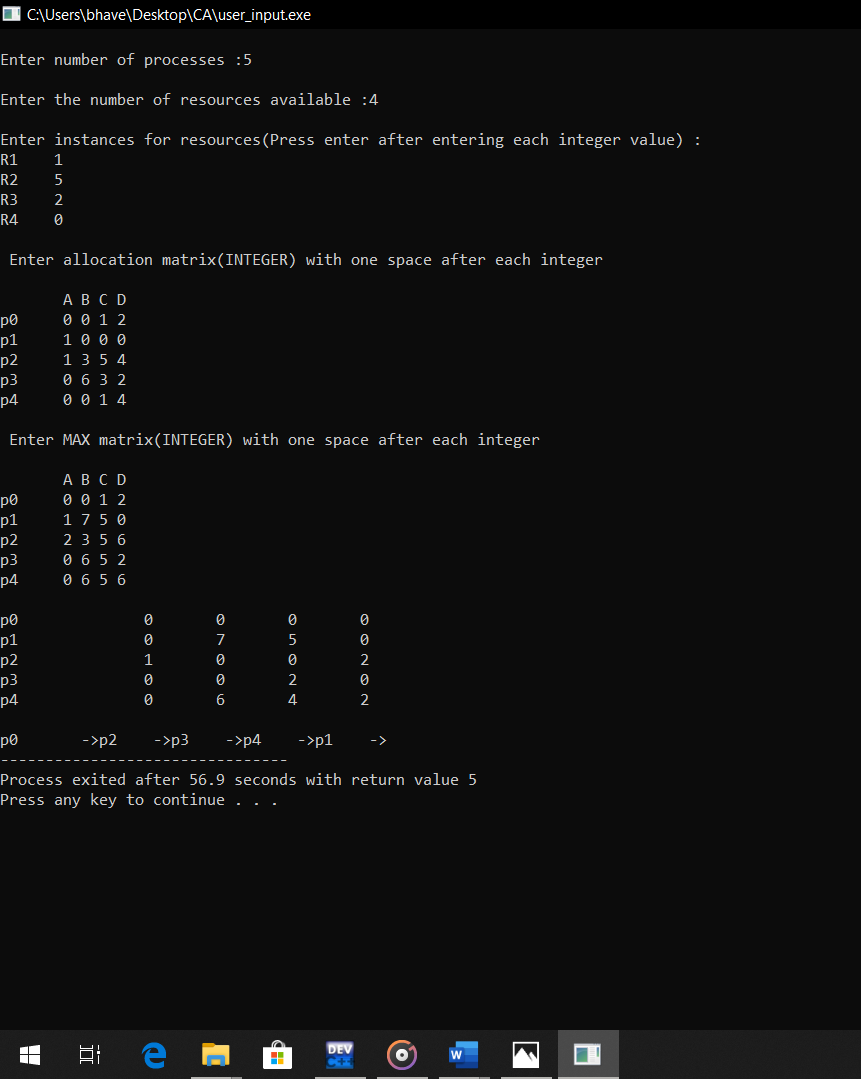




**Output:-**

**Code:-(User is asked to enter the values)**

****

****