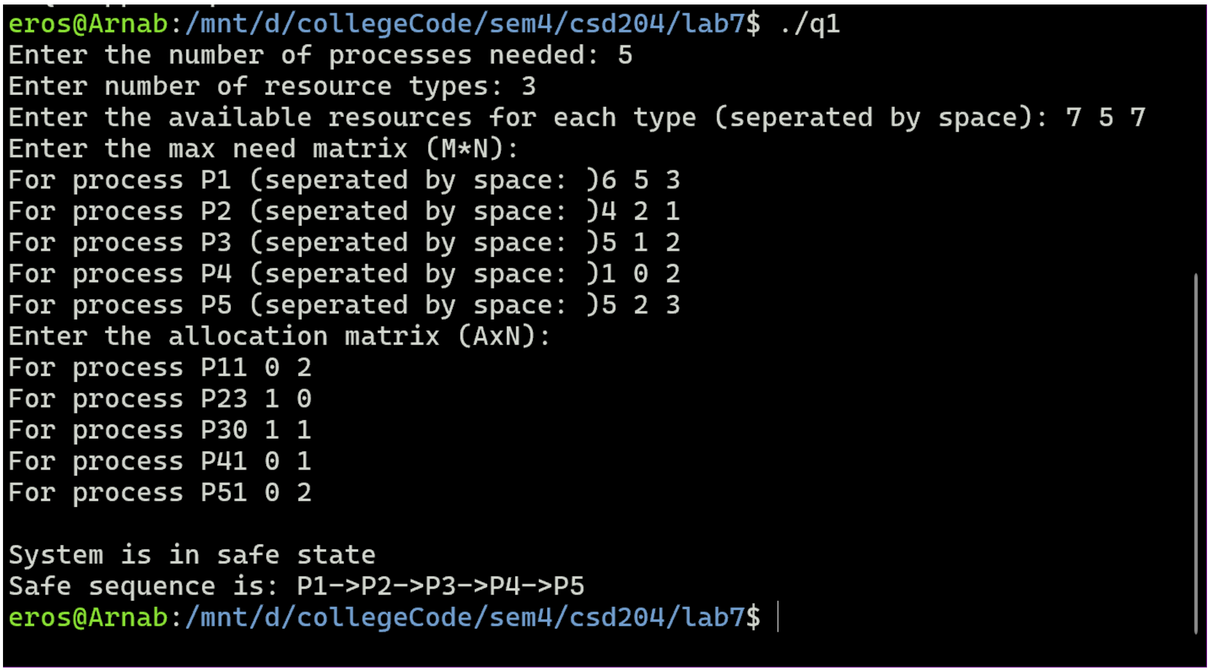
Q1)



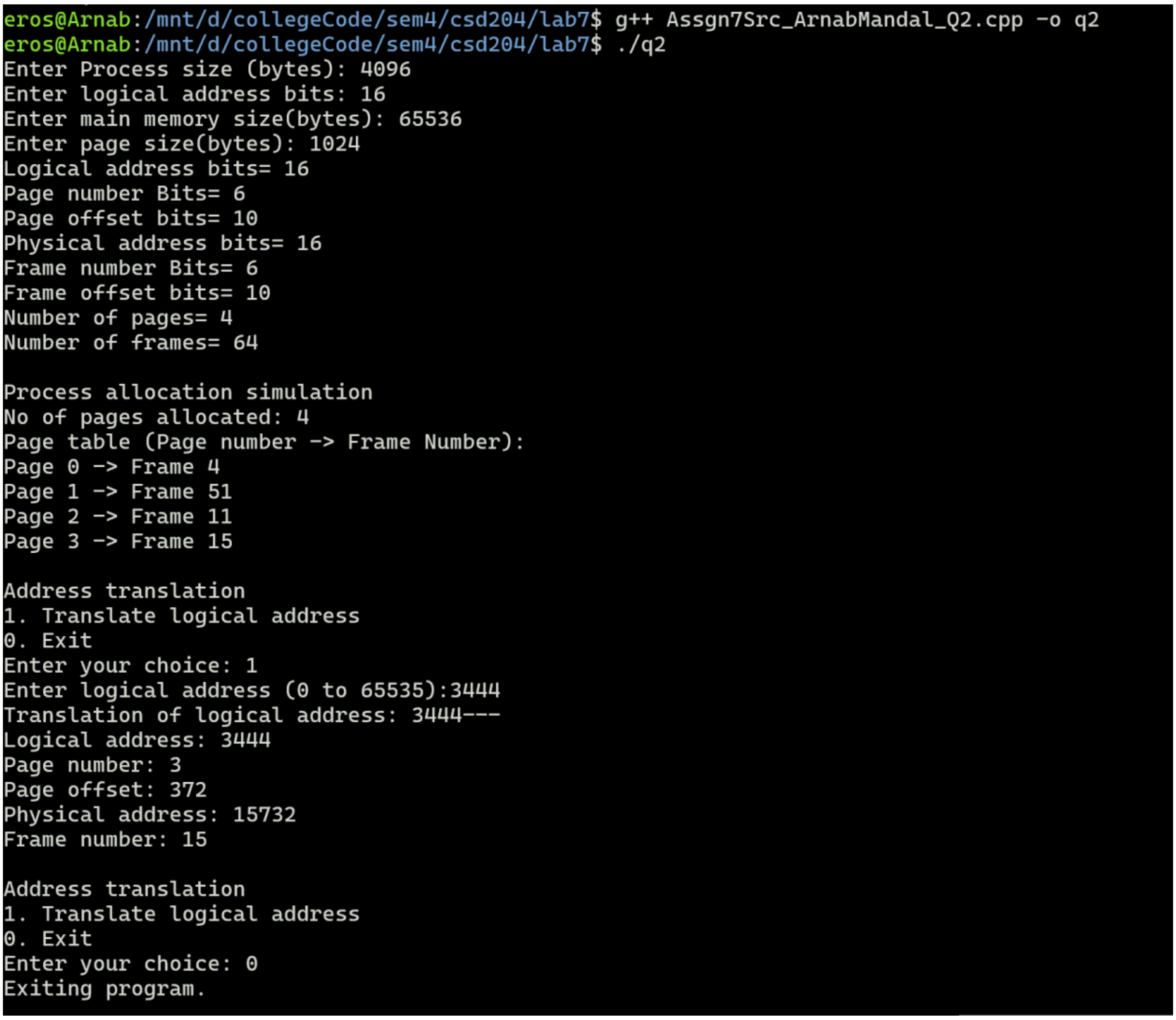
```
eros@Arnab:/mnt/d/collegeCode/sem4/csd204/lab7$ ./q1
Enter the number of processes needed: 5
Enter number of resource types: 3
Enter the available resources for each type (seperated by space): 7 5 7
Enter the max need matrix (M*N):
For process P1 (seperated by space: )6 5 3
For process P2 (seperated by space: )4 2 1
For process P3 (seperated by space: )5 1 2
For process P4 (seperated by space: )1 0 2
For process P5 (seperated by space: )5 2 3
Enter the allocation matrix (AxN):
For process P11 0 2
For process P23 1 0
For process P30 1 1
For process P41 0 1
For process P51 0 2

System is in safe state
Safe sequence is: P1->P2->P3->P4->P5
eros@Arnab:/mnt/d/collegeCode/sem4/csd204/lab7$
```

Successful carry out of trial cases

Note multiple safe sequences can exist as long as they don’t form a deadlock causing chain

Q2)



```
eros@Arnab:/mnt/d/collegeCode/sem4/csd204/lab7$ g++ Assgn7Src_ArnabMandal_Q2.cpp -o q2
eros@Arnab:/mnt/d/collegeCode/sem4/csd204/lab7$ ./q2
Enter Process size (bytes): 4096
Enter logical address bits: 16
Enter main memory size(bytes): 65536
Enter page size(bytes): 1024
Logical address bits= 16
Page number Bits= 6
Page offset bits= 10
Physical address bits= 16
Frame number Bits= 6
Frame offset bits= 10
Number of pages= 4
Number of frames= 64

Process allocation simulation
No of pages allocated: 4
Page table (Page number -> Frame Number):
Page 0 -> Frame 4
Page 1 -> Frame 51
Page 2 -> Frame 11
Page 3 -> Frame 15

Address translation
1. Translate logical address
0. Exit
Enter your choice: 1
Enter logical address (0 to 65535):3444
Translation of logical address: 3444---
Logical address: 3444
Page number: 3
Page offset: 372
Physical address: 15732
Frame number: 15

Address translation
1. Translate logical address
0. Exit
Enter your choice: 0
Exiting program.
```

Successful interactive translation and mathematical conversion as requested in question