Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA   Batch - 41
OS Practical 8

1. **Create any two Resource Allocation Graphs which may lead to deadlock and system in unsafe state of your choice. Demonstrate the RAGs using C code. (Hint : Sample RAG as Instructed by Instructor)**

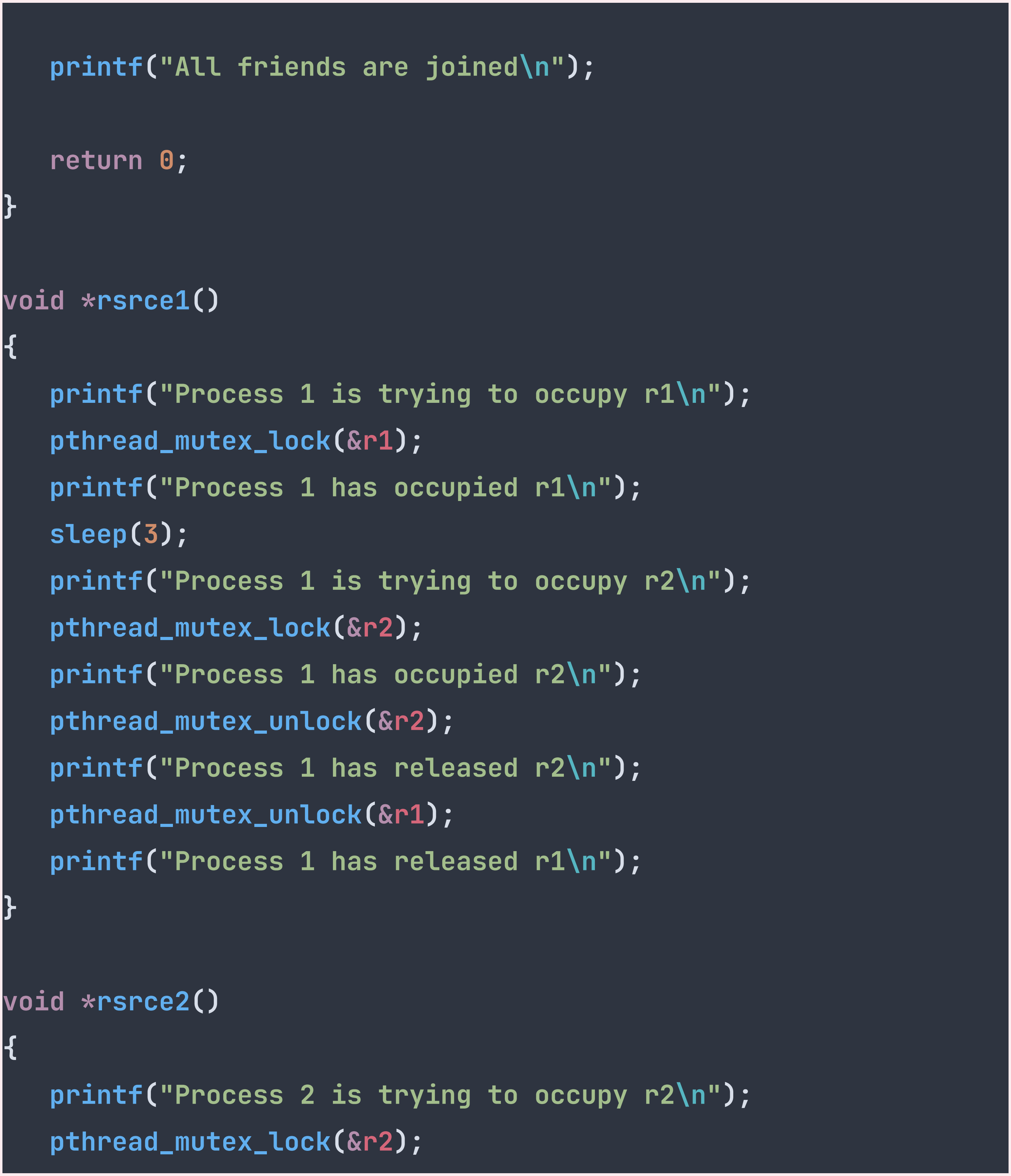
Code :

```
#include<stdio.h>
#include<pthread.h>
#include<unistd.h>

void *rsrce1();
void *rsrce2();
void *rsrce3();
pthread_mutex_t r1, r2, r3;

int main()
{
   pthread_mutex_init(&r1, NULL);
   pthread_mutex_init(&r2, NULL);
   pthread_mutex_init(&r3, NULL);
   pthread_t P1, P2, P3;
   pthread_create(&P1, NULL, rsrce1, NULL);
   pthread_create(&P2, NULL, rsrce2, NULL);
   pthread_create(&P3, NULL, rsrce3, NULL);
   pthread_join(P1, NULL);
   pthread_join(P2, NULL);
   pthread_join(P3, NULL);
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA   Batch - 41
OS Practical 8

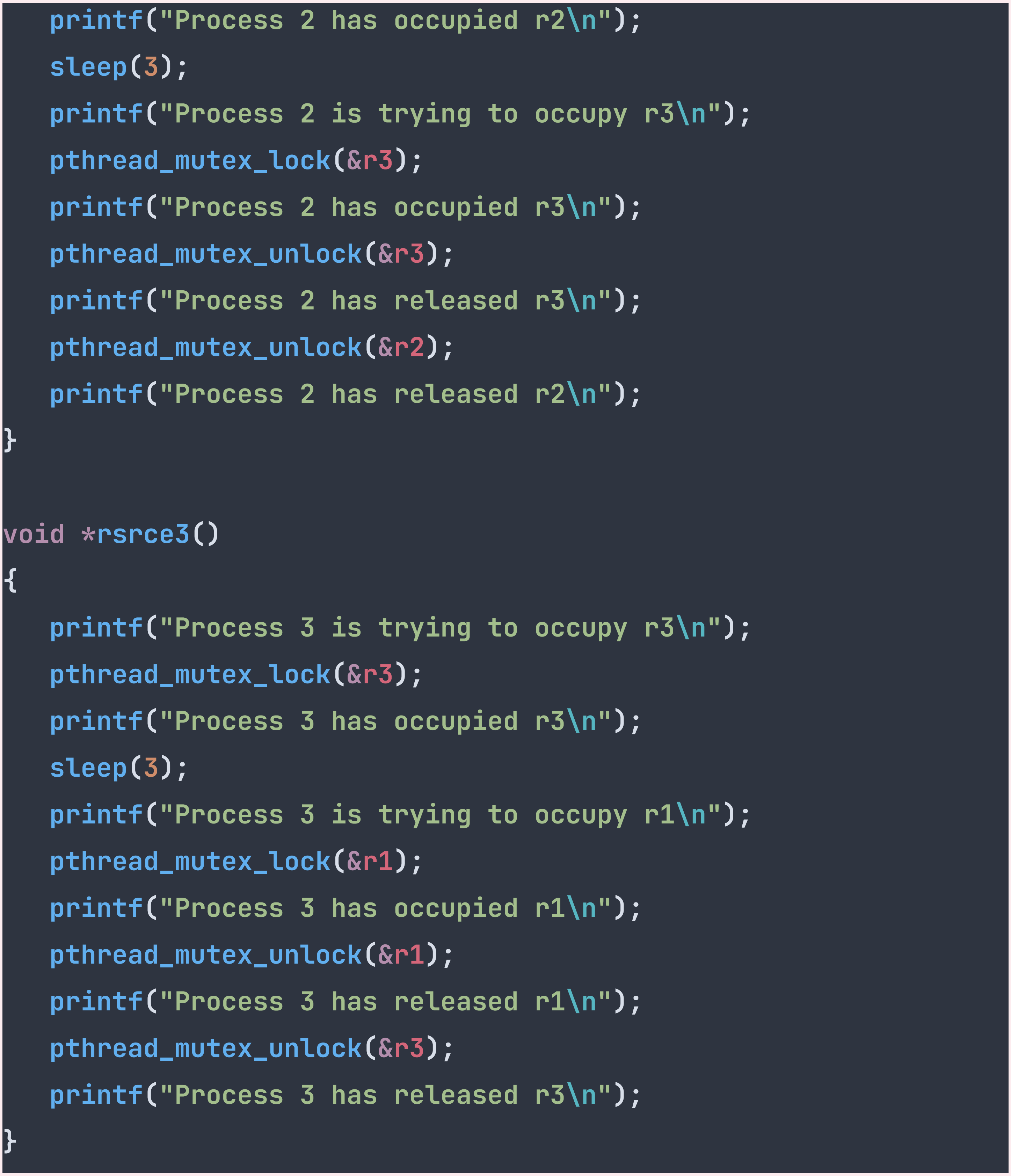


```
   printf("All friends are joined\n");

   return 0;
}

void *rsrce1()
{
   printf("Process 1 is trying to occupy r1\n");
   pthread_mutex_lock(&r1);
   printf("Process 1 has occupied r1\n");
   sleep(3);
   printf("Process 1 is trying to occupy r2\n");
   pthread_mutex_lock(&r2);
   printf("Process 1 has occupied r2\n");
   pthread_mutex_unlock(&r2);
   printf("Process 1 has released r2\n");
   pthread_mutex_unlock(&r1);
   printf("Process 1 has released r1\n");
}

void *rsrce2()
{
   printf("Process 2 is trying to occupy r2\n");
   pthread_mutex_lock(&r2);
```

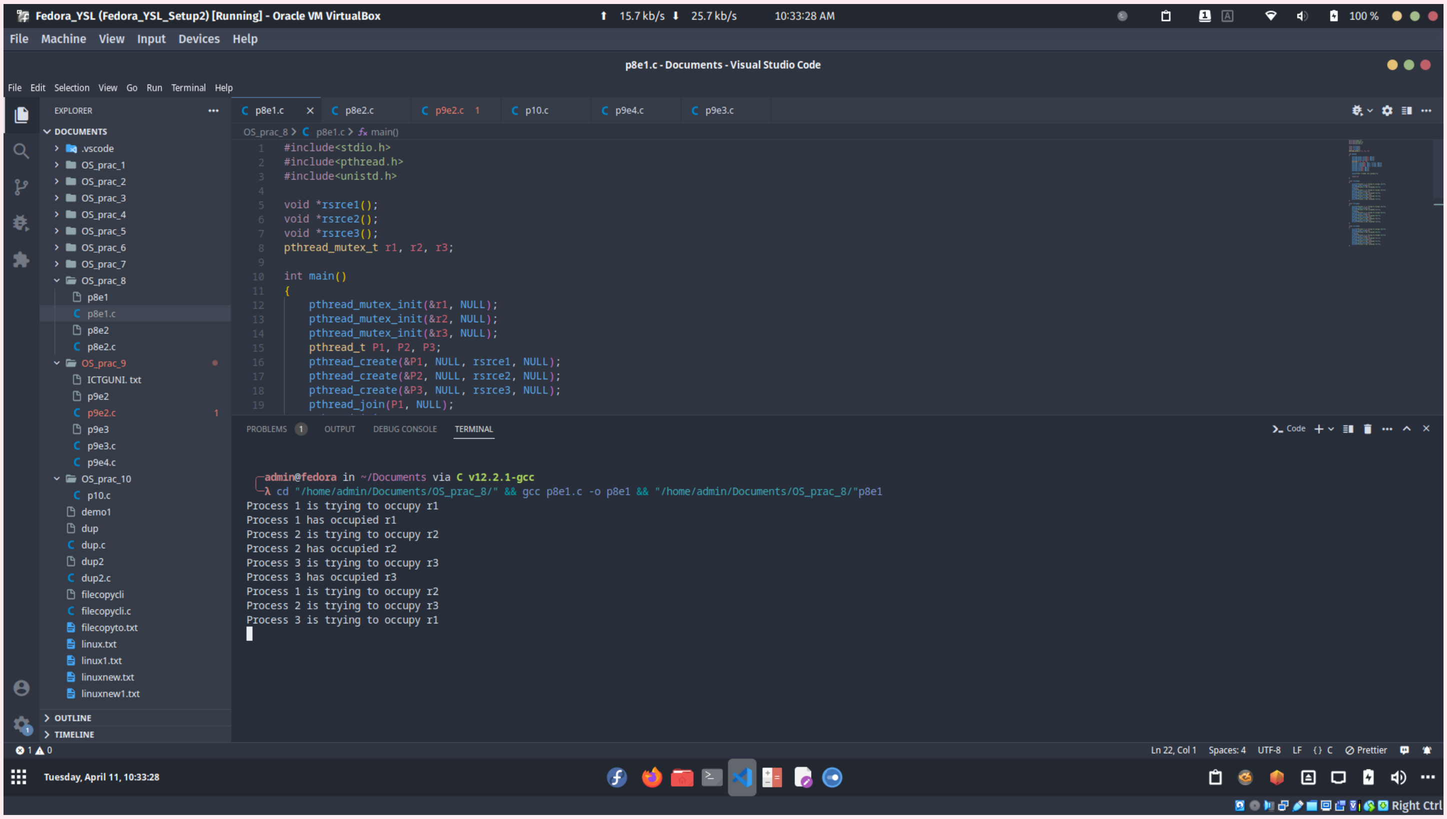


```
    printf("Process 2 has occupied r2\n");
    sleep(3);
    printf("Process 2 is trying to occupy r3\n");
    pthread_mutex_lock(&r3);
    printf("Process 2 has occupied r3\n");
    pthread_mutex_unlock(&r3);
    printf("Process 2 has released r3\n");
    pthread_mutex_unlock(&r2);
    printf("Process 2 has released r2\n");
}

void *rsrce3()
{
    printf("Process 3 is trying to occupy r3\n");
    pthread_mutex_lock(&r3);
    printf("Process 3 has occupied r3\n");
    sleep(3);
    printf("Process 3 is trying to occupy r1\n");
    pthread_mutex_lock(&r1);
    printf("Process 3 has occupied r1\n");
    pthread_mutex_unlock(&r1);
    printf("Process 3 has released r1\n");
    pthread_mutex_unlock(&r3);
    printf("Process 3 has released r3\n");
}
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA   Batch - 41
OS Practical 8

Output :


Fedora_YSL (Fedora_YSL_Setup2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
p8e1.c - Documents - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
p8e1.c
p8e2.c
p9e2.c
p10.c
p9e4.c
p9e3.c
OS_prac_8 > p8e1.c > main()
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
Tuesday, April 11, 10:33:28

```c
#include<stdio.h>
#include<pthread.h>
#include<unistd.h>

void *rsrce1();
void *rsrce2();
void *rsrce3();
pthread_mutex_t r1, r2, r3;

int main()
{
    pthread_mutex_init(&r1, NULL);
    pthread_mutex_init(&r2, NULL);
    pthread_mutex_init(&r3, NULL);
    pthread_t P1, P2, P3;
    pthread_create(&P1, NULL, rsrce1, NULL);
    pthread_create(&P2, NULL, rsrce2, NULL);
    pthread_create(&P3, NULL, rsrce3, NULL);
    pthread_join(P1, NULL);
```

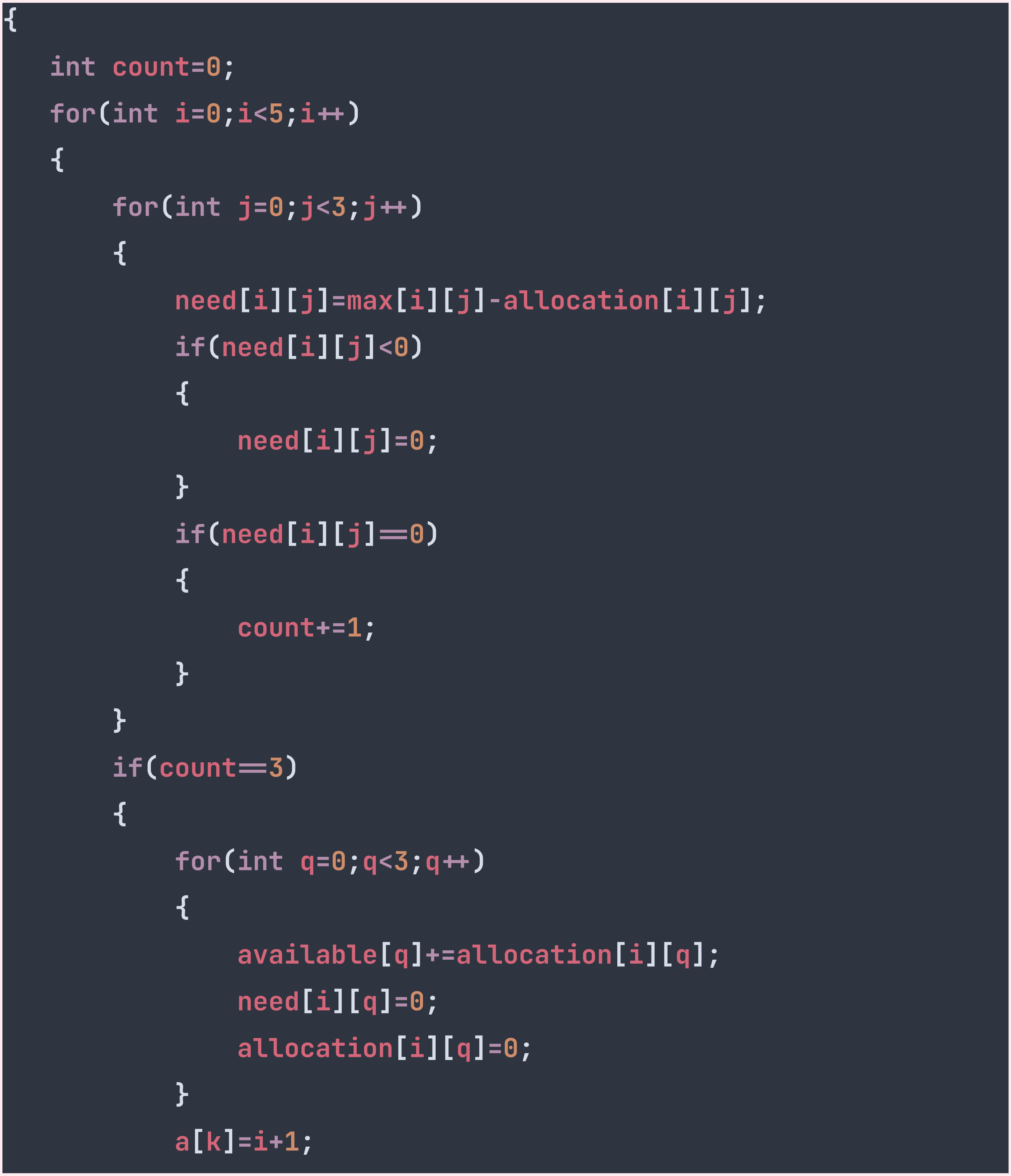
```
╭─admin@fedora in ~/Documents via C v12.2.1-gcc
╰─λ cd "/home/admin/Documents/OS_prac_8/" && gcc p8e1.c -o p8e1 && "/home/admin/Documents/OS_prac_8/"p8e1
Process 1 is trying to occupy r1
Process 1 has occupied r1
Process 2 is trying to occupy r2
Process 2 has occupied r2
Process 3 is trying to occupy r3
Process 3 has occupied r3
Process 1 is trying to occupy r2
Process 2 is trying to occupy r3
Process 3 is trying to occupy r1
```

2. **For a given Problem -1 , Design the solution after applying Banker Algorithm using C program.**

Code :

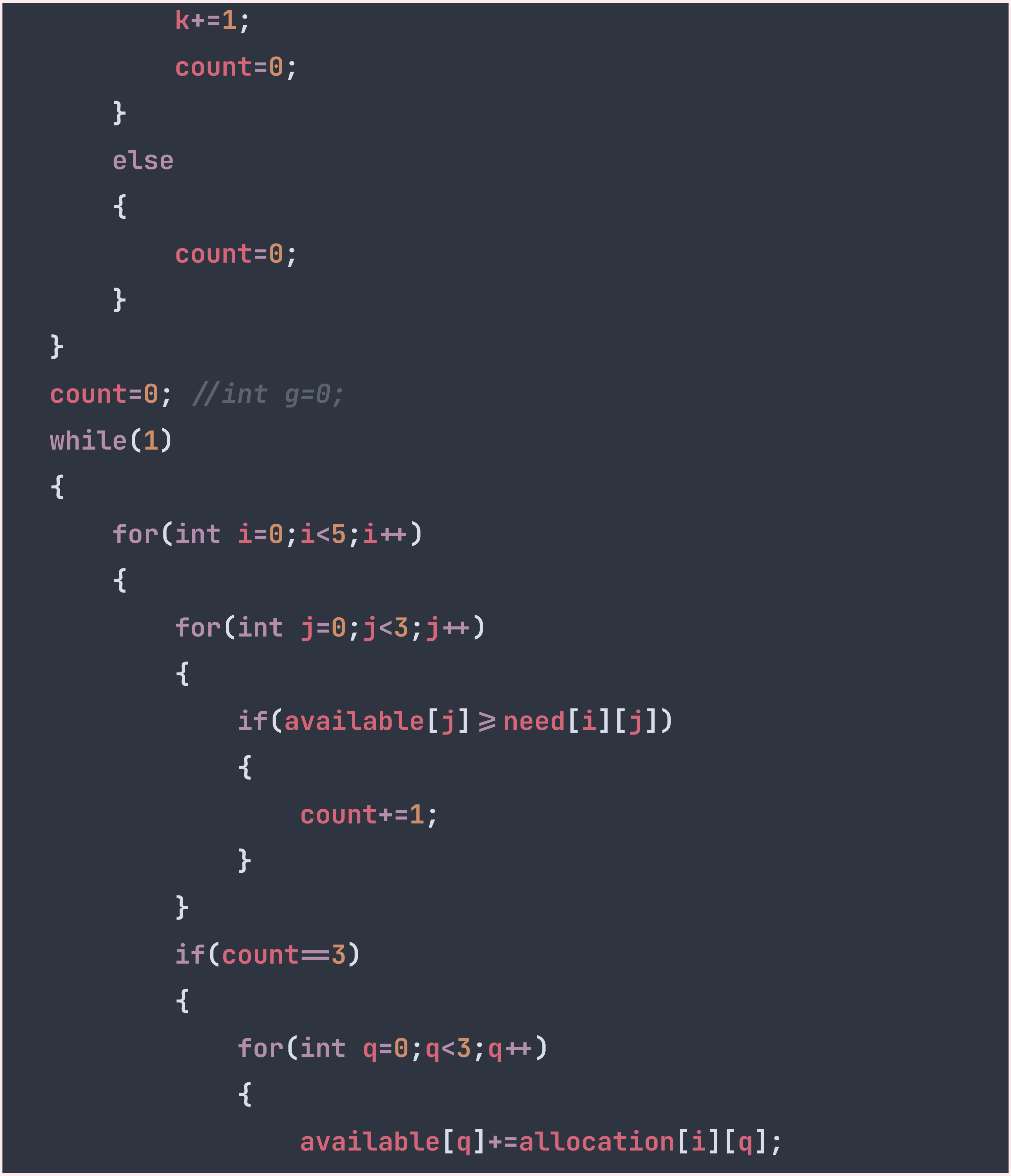
```c
#include"stdio.h"
int allocation[5][3]={{1,1,2},{2,1,2},{4,0,1},{0,2,0},{1,1,2}};
int max[5][3]={{4,3,3},{3,2,2},{9,0,2},{7,5,3},{1,1,2}};
//dependent
int need[5][3]={0}; int available[3] = {2,1,0}; //dependent
int a[5]={0}; int k=0; int maxA=10,maxB=6,maxC=7; //dependent
int main()
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA   Batch - 41
OS Practical 8

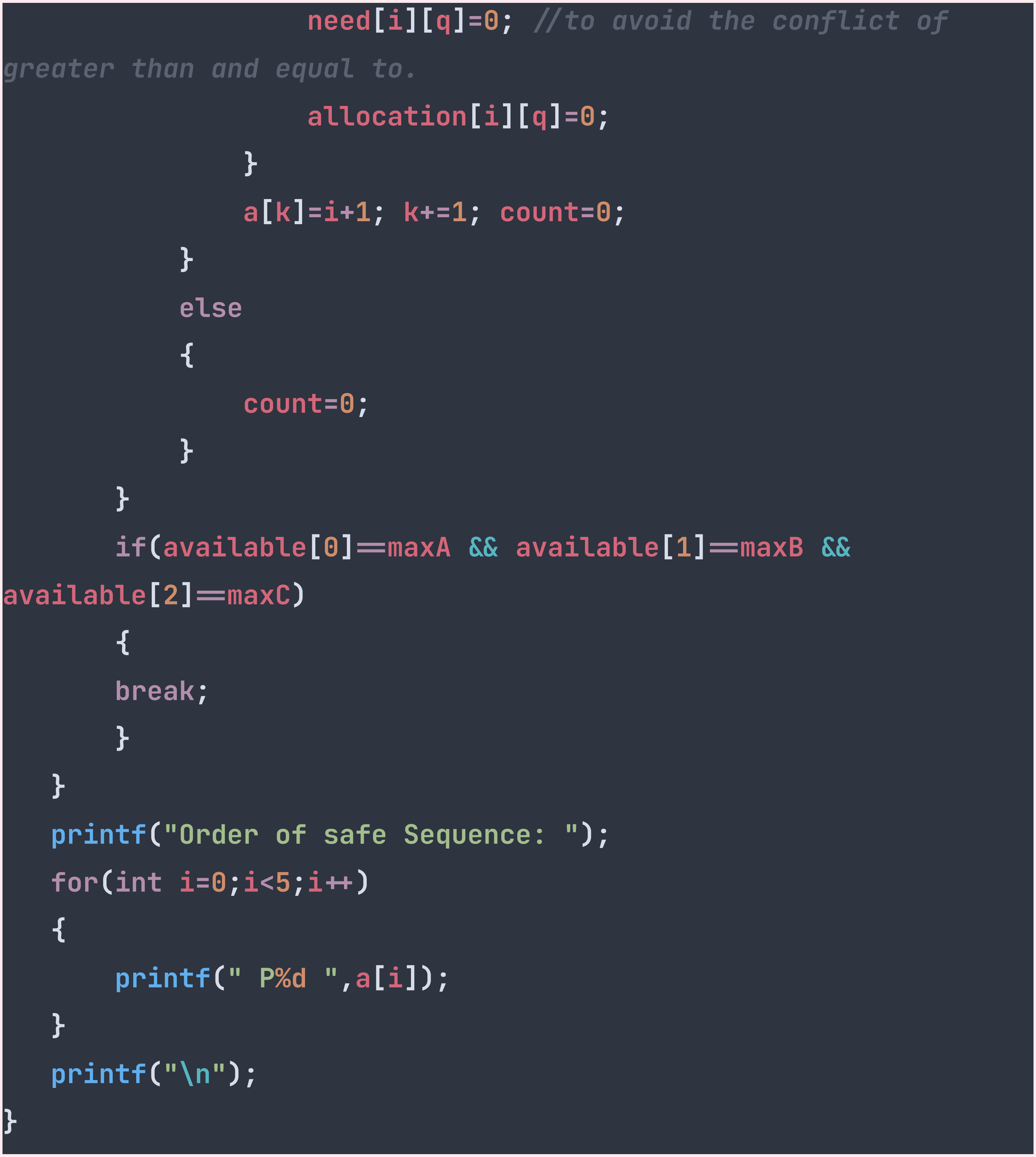


```
{
   int count=0;
   for(int i=0;i<5;i++)
   {
       for(int j=0;j<3;j++)
       {
           need[i][j]=max[i][j]-allocation[i][j];
           if(need[i][j]<0)
           {
               need[i][j]=0;
           }
           if(need[i][j]==0)
           {
               count+=1;
           }
       }
       if(count==3)
       {
           for(int q=0;q<3;q++)
           {
               available[q]+=allocation[i][q];
               need[i][q]=0;
               allocation[i][q]=0;
           }
           a[k]=i+1;
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA   Batch - 41
OS Practical 8



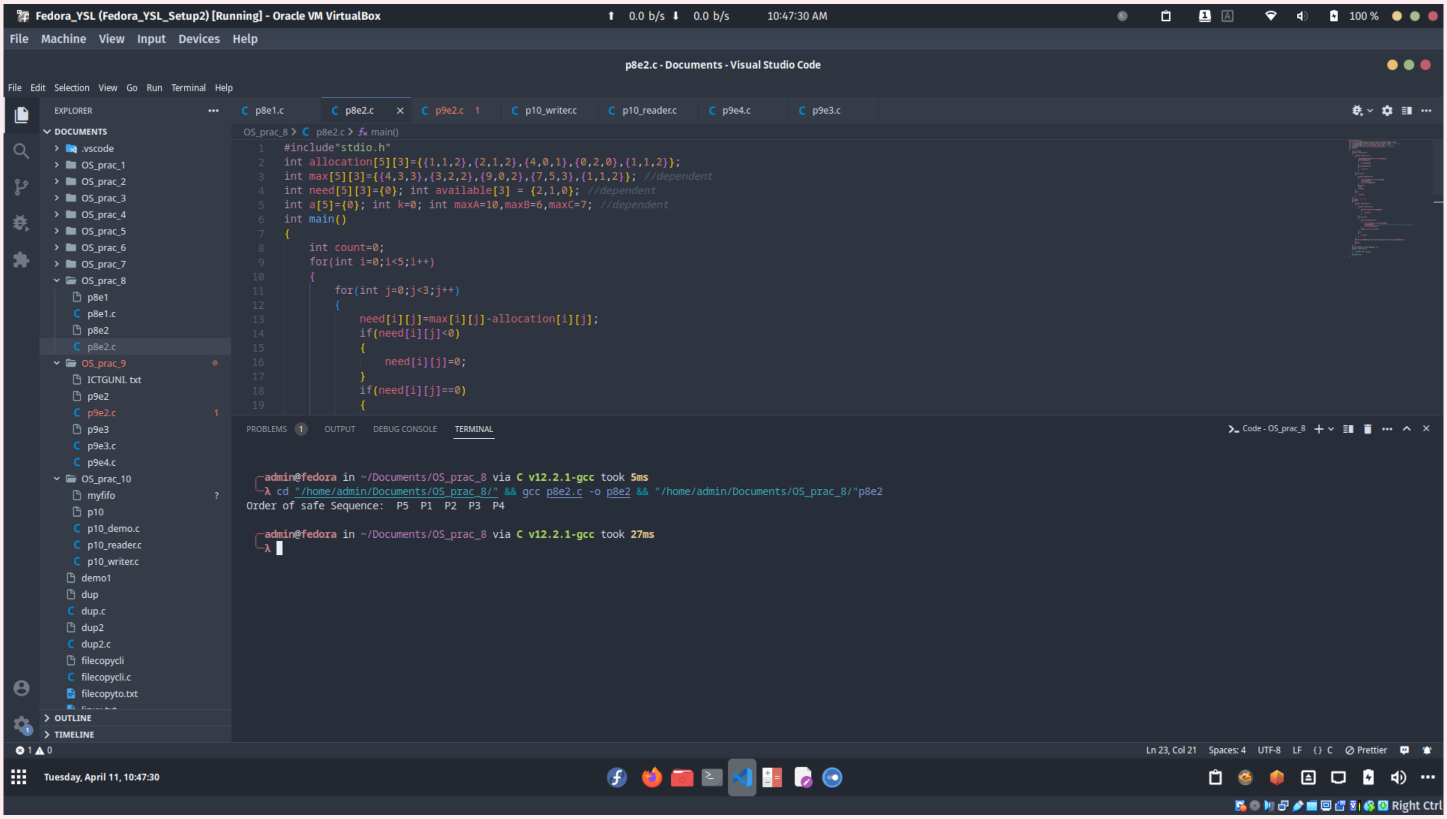
```
            k+=1;
            count=0;
        }
        else
        {
            count=0;
        }
    }
    count=0; //int g=0;
    while(1)
    {
        for(int i=0;i<5;i++)
        {
            for(int j=0;j<3;j++)
            {
                if(available[j]>=need[i][j])
                {
                    count+=1;
                }
            }
            if(count==3)
            {
                for(int q=0;q<3;q++)
                {
                    available[q]+=allocation[i][q];
```



```
                    need[i][q]=0; //to avoid the conflict of greater than and equal to.
                    allocation[i][q]=0;
                }
                a[k]=i+1; k+=1; count=0;
            }
            else
            {
                count=0;
            }
        }
        if(available[0]==maxA && available[1]==maxB && available[2]==maxC)
        {
        break;
        }
    }
    printf("Order of safe Sequence: ");
    for(int i=0;i<5;i++)
    {
        printf(" P%d ",a[i]);
    }
    printf("\n");
}
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA   Batch - 41
OS Practical 8

Output :



```c
#include"stdio.h"
int allocation[5][3]={{1,1,2},{2,1,2},{4,0,1},{0,2,0},{1,1,2}};
int max[5][3]={{4,3,3},{3,2,2},{9,0,2},{7,5,3},{1,1,2}}; //dependent
int need[5][3]={0}; int available[3] = {2,1,0}; //dependent
int a[5]={0}; int k=0; int maxA=10,maxB=6,maxC=7; //dependent
int main()
{
    int count=0;
    for(int i=0;i<5;i++)
    {
        for(int j=0;j<3;j++)
        {
            need[i][j]=max[i][j]-allocation[i][j];
            if(need[i][j]<0)
            {
                need[i][j]=0;
            }
            if(need[i][j]==0)
            {
```

```
╭─admin@fedora in ~/Documents/OS_prac_8 via C v12.2.1-gcc took 5ms
╰─λ cd "/home/admin/Documents/OS_prac_8/" && gcc p8e2.c -o p8e2 && "/home/admin/Documents/OS_prac_8/"p8e2
Order of safe Sequence:  P5  P1  P2  P3  P4

╭─admin@fedora in ~/Documents/OS_prac_8 via C v12.2.1-gcc took 27ms
╰─λ
```

Fedora_YSL (Fedora_YSL_Setup2) [Running] - Oracle VM VirtualBox
10:47:30 AM
File Machine View Input Devices Help
p8e2.c - Documents - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
DOCUMENTS
p8e1.c
p8e2.c
p9e2.c
p10_writer.c
p10_reader.c
p9e4.c
p9e3.c
OS_prac_8 > p8e2.c > main()
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
Ln 23, Col 21 Spaces: 4 UTF-8 LF C Prettier
Tuesday, April 11, 10:47:30
Right Ctrl