Script started on 2020-03-29 20:28:53+0530
GAYU@GAYU: ~/Desktop/bank;
GAYU@GAYU [00m: [01;34m~/Desktop/bank $gcc banker.c -o bank

GAYU@GAYU~/Desktop/bank $ cat banker.c

```
#include<stdio.h>
#include<stdlib.h>
int no_of_processes;
int no_of_resources;
void print(int process[10],int allocated[no_of_processes][no_of_resources],int max[no_of_processes][no_of_resourc
es], int need[no_of_processes][no_of_resources], int available[no_of_resources])
{
 printf("\nPID\t\tAllocation\t\tMaximum\t\tNeed\t\tAvailable\n");
 printf("   \t\tA  B  C   \t\tA B C  \t\tA B C\t\tA  B  C\n");
 for(int i=0;i<no_of_processes;i++)
 {
  printf("P%d \t\t",process[i]);
  for(int j=0;j<no_of_resources;j++)
   printf("%d  ",allocated[i][j]);
  printf("   \t\t");

  for(int j=0;j<no_of_resources;j++)
   printf("%d ",max[i][j]);
  printf("  \t");

  for(int j=0;j<no_of_resources;j++)
   printf("%d ",need[i][j]);
  printf("\t\t");
  for(int j=0;j<no_of_resources;j++)
   printf("%d  ",available[j]);
  printf("\n");
 }
}
int bankers(int process[10],int allocated[no_of_processes][no_of_resources],int max[no_of_processes][no_of_resour
ces], int need[no_of_processes][no_of_resources], int available[no_of_resources])
{
 int seq[no_of_processes];
 int work[no_of_resources];
 int finish[no_of_processes];
 int check;
 int ind = 0;
 for(int i=0;i<no_of_resources;i++)
 {
  work[i] = available[i];
 }
 for(int i = 0;i<no_of_processes;i++)
  finish[i] =0;
 int m = 0;
 for(int k=0;k<no_of_processes;k++)  {
  for(int i=0;i<no_of_processes;i++)  {
   if(finish[i]==0)  {
    int flag = 0;
    for(int j=0;j<no_of_resources;j++) {
     if(need[i][j] > work[j]) {
```

```
      flag =1 ;
      break;
     }
    }
    if(flag == 0) {
     seq[ind] = i;
     for(m=0;m<no_of_resources;m++)
        work[m] += allocated[i][m];
     finish[i] = 1;
     ind++;
    }
   }
  }
 }
 int check_safe = 0;
 for(int i=0;i<no_of_processes;i++)
 {
  if(finish[i] == 0)
  {
   check_safe = 1;
   break;
  }
 }
 if(check_safe == 0)
  {
   printf("System is in safe state \n");
   for(int i=0;i<no_of_processes;i++)
    printf("P%d ->",seq[i]);
    print(process,allocated,max,need,available);
    return 1;
  }
 else
  printf("System is in unsafe state \n");
 return 0;

}
void request(int process[10],int allocated[no_of_processes][no_of_resources],int max[no_of_processes][no_of_reso
urces], int need[no_of_processes][no_of_resources], int available[no_of_resources])
{
 printf("Enter procees id and request \n");
 int req_p;
 int request[no_of_resources];
 scanf("%d",&req_p);
 for(int i=0;i<no_of_resources;i++)
  scanf("%d",&request[i]);
 int check_need = 0;
 int check_avail = 0;
 for(int i=0;i<no_of_resources;i++)
 {
  if(request[i] > need[req_p][i])
   { check_need = 1;
     break;
          }
 }
```

```
 if(check_need == 0)
 {
  for(int j=0;j<no_of_resources;j++)
  {
   if(request[j] > available[j])
    {
    check_avail = 1;
    break;
    }
  }

  if(check_avail == 0)
  {
   for(int i=0;i<no_of_resources;i++)
   {
    allocated[req_p][i] += request[i];
    need[req_p][i] -= request[i];
    available[i] -= request[i];
   }
   int c = bankers(process,allocated,max, need, available);
   if(c == 0)
    {
   for(int i=0;i<no_of_resources;i++)
    {
    allocated[req_p][i] -= request[i];
    need[req_p][i] += request[i];
    available[i] += request[i];
   }
   }
  }
  else
   printf("Resources not available \n");
 }
 else
  printf("Request exceeding claim . Process HALTED\n");
}
void main()
{
 printf("Enter number of processes ");
 scanf("%d",&no_of_processes);
 int process[no_of_processes];
 for(int i=0;i<no_of_processes;i++)
  process[i] = i;
 printf("\nEnter number of resources:");
 scanf("%d",&no_of_resources);
 int resource[no_of_resources];
 for(int i=0;i<no_of_resources;i++)
  resource[i]=i;
 int available[no_of_resources];
 for(int i=0;i<no_of_resources;i++)
 {
  printf("Enter Number of available instances of%d:  ",resource[i]);
  scanf("%d",&available[i]);
 }
```

```
 int max[no_of_processes][no_of_resources];
 for(int i=0;i<no_of_processes;i++)
 {
  printf("Enter Maximum Requirement for P%d:  ",process[i]);
  for(int j=0;j<no_of_resources;j++)
   scanf("%d",&max[i][j]);
 }
 int allocated[no_of_processes][no_of_resources];
 for(int i=0;i<no_of_processes;i++)
 {
   printf("Enter Allocated instances to P%d:  ",process[i]);
  for(int j=0;j<no_of_resources;j++)
   scanf("%d",&allocated[i][j]);
 }
 int need[no_of_processes][no_of_resources];
 for(int i=0;i<no_of_processes;i++)
 {
  for(int j=0;j<no_of_resources;j++)
  need[i][j] = max[i][j] - allocated[i][j];
 }

 int choice;
 do
 {
  printf("1.Bankers \n");
  printf("2.Request \n");
  printf("3.Exit \n");
  printf("Enter choice \n");
  scanf("%d",&choice);
  switch(choice)
  {
   case 1: bankers(process,allocated,max, need, available);
    break;
   case 2: request(process, allocated, max, need, available);
    break;
  }
 }while(choice!=3);
}

 ]0;GAYU@GAYU: ~/Desktop/bank  [01;32mGAYU@GAYU [00m: [01;34m~/Desktop/bank [00m$ cat banker.c
        gcc banker.c -o bank                    ./c [K  [Kbank
Enter number of processes 5

Enter number of resources:3
Enter Number of available instances of0:  3
Enter Number of available instances of1:  3
Enter Number of available instances of2:  2
Enter Maximum Requirement for P0:  7 5 3
Enter Maximum Requirement for P1:  32    2 2
Enter Maximum Requirement for P2:  9 0 2
Enter Maximum Requirement for P3:  2 2 2
Enter Maximum Requirement for P4:  4 3 3
Enter Allocated instances to P0:  0 1 0
Enter Allocated instances to P1:  2 0 0
```

```
Enter Allocated instances to P2:  3 02        2
Enter Allocated instances to P3:  2 1 1
Enter Allocated instances to P4:  -  -           0 0 2
1.Bankers
2.Request
3.Exit
Enter choice
1
System is in safe state
P1 ->P3 ->P4 ->P0 ->P2 ->
PID  Allocation  Maximum  Need  Available
     A  B  C     A B C    A B C  A  B  C
P0   0  1  0      7 5 3    7 4 3   3  3  2
P1   2  0  0      3 2 2    1 2 2   3  3  2
P2   3  0  2      9 0 2    6 0 0   3  3  2
P3   2  1  1      2 2 2    0 1 1   3  3  2
P4   0  0  2      4 3 3    4 3 1   3  3  2
1.Bankers
2.Request
3.Exit
Enter choice
2
Enter procees id and request
1
1 0 2
System is in safe state
P1 ->P3 ->P4 ->P0 ->P2 ->
PID  Allocation  Maximum  Need  Available
     A  B  C     A B C    A B C  A  B  C
P0   0  1  0      7 5 3    7 4 3   2  3  0
P1   3  0  2      3 2 2    0 2 0   2  3  0
P2   3  0  2      9 0 2    6 0 0   2  3  0
P3   2  1  1      2 2 2    0 1 1   2  3  0
P4   0  0  2      4 3 3    4 3 1   2  3  0
1.Bankers
2.Request
3.Exit
Enter choice
1
System is in safe state
P1 ->P3 ->P4 ->P0 ->P2 ->
PID  Allocation  Maximum  Need  Available
     A  B  C     A B C    A B C  A  B  C
P0   0  1  0      7 5 3    7 4 3   2  3  0
P1   3  0  2      3 2 2    0 2 0   2  3  0
P2   3  0  2      9 0 2    6 0 0   2  3  0
P3   2  1  1      2 2 2    0 1 1   2  3  0
P4   0  0  2      4 3 3    4 3 1   2  3  0
1.Bankers
2.Request
3.Exit
Enter choice
2
Enter procees id and request
```

```
0
3 3 4
Request exceeding claim . Process HALTED
1.Bankers
2.Request
3.Exit
Enter choice
3
 ]0;GAYU@GAYU: ~/Desktop/bank  [01;32mGAYU@GAYU [00m: [01;34m~/Desktop/bank [00m$ ei  [Kxit
exit

Script done on 2020-03-29 20:31:22+0530
```