```
/*
 ============================================================================
 Name        : os_2.c
 Author      :mostafa  elgohary
 Version     :
 Copyright   : Your copyright notice
 Description : Hello World in C, Ansi-style
 ============================================================================
 */

#include <stdio.h>
#include <stdlib.h>



typedef enum { false, true } bool;

int main(void) {

        int num_process,num_resourses;
        int i,j,*ptr;
    int finish;
    bool check_available=true;
    int new_process;



    printf("enter the number of process :  ");
    scanf("%d",&num_process);

    printf("enter the number of available  resourses:  ");
     scanf("%d",&num_resourses);




        finish=num_process;

        int check [num_process];
     int new_process_arr[num_resourses];

        for(i=0;i<num_process;i++){

                check[i]=0;

        }

        int max [num_process][num_resourses];
        int allocation [num_process][num_resourses];
        int need [num_process][num_resourses];
        int  available[num_process+1][num_resourses];







        /*********************************************************/

        /// input  allocation array

        printf("\n\t enter the allocation array \n");
         for(i=0;i<num_process;i++){
```

```
                printf(" enter the allocation of process %d \n",i+1);
                        for(j=0;j<num_resourses;j++)
                                scanf("%d",&allocation[i][j]);
        }



        //// check  allocate  the allocation array
        printf(" \n you  entered this array \n");
              for(i=0;i<num_process;i++){
                                        for(j=0;j<num_resourses;j++)
                                                printf("%-8d",allocation[i][j]);
                                printf("\n");
              }


/*********************************************************************/

                /// input  max array

                printf("\n\t enter the max array \n");
                 for(i=0;i<num_process;i++){

                        printf(" enter the allocation of process %d \n",i+1);
                                for(j=0;j<num_resourses;j++)
                                        scanf("%d",&max[i][j]);
                 }



                 //// check  allocate  the max array
                        printf(" \n you  entered this array \n");
                              for(i=0;i<num_process;i++){

for(j=0;j<num_resourses;j++)

printf("%-8d",max[i][j]);
                                                        printf("\n");
                                }


/*********************************************************************/

         //  input  the available  resoures

          printf("\n enter the available  instance of each  resouses ");

                              for(j=0;j<num_resourses;j++)
                                  scanf("%d",&available[0][j]);




/*********************************************************************/

                              printf(" \n   this is the need  array   \n");

                              for(i=0;i<num_process;i++){

                                   for(j=0;j<num_resourses;j++){

                                          need[i][j]=max[i][j]-allocation[i][j];
                                        printf("%-8d",need[i][j]);

                                   }
```

```
                                                    printf("\n");
                                }

 //*********************************************************************//
                                //  check the stabitty



while(finish!=0){

    for(i=0;i<num_process;i++){

       check_available=true;
          if(check[i]==1){
              printf(" the check not %d\n",i);

                  continue;
          }

        for(j=0;j<num_resourses;j++){

                if(available[i][j]<need[i][j]){
                 check_available=false;
                 printf(" \n the process %d not avaliabe \n ",i);
                 break;
                        //  available[i+1][j]=available[i][j]+allocation[i][j];
                }
        }

     printf("\n the available now is %d :  ",i);


      for(j=0;j<num_resourses;j++)      {

          if(check_available==true){
                check[i]=1;  // to don't repeat

                  available[i+1][j]=available[i][j]+allocation[i][j];
                  printf("   %d",available[i+1][j]);
          }


      else{
          available[i+1][j]=available[i][j];
            printf(" %d",available[i][j]);
      }

      }

  if(check[i]==1)

  printf("\n\n process P %d done ",i);

  finish--;


         }

}


 //*********************************************************************///
```

```
    printf("\nenter the number of  new process that arrived  \n");
    scanf("%d",&new_process);

    printf("enter the allocation ");

    for(i=0;i<num_resourses;i++){
    scanf("%d",&new_process_arr[i]);
     allocation[new_process][i]+=new_process_arr[i];
     available[0][i]-=new_process_arr[i];

    }




    /**************************************************************************/
                               printf(" \n   this is the  new need  array   \n\n");

                               for(i=0;i<num_process;i++){

                                    for(j=0;j<num_resourses;j++){

                                           need[i][j]=max[i][j]-allocation[i][j];
                                         printf("%-8d",need[i][j]);

                                    }
                                                               printf("\n");
                               }

   //************************************************************************//

     // check  the stabity of new  input

    finish=num_process;
    for(i=0;i<num_process;i++)
      check[i]=0;


                               while(finish!=0){

      for(i=0;i<num_process;i++){

          check_available=true;
            if(check[i]==1){

            continue;
           }

          for(j=0;j<num_resourses;j++){

                  if(available[i][j]<need[i][j]){
                   check_available=false;
                   printf(" \n the process %d not avaliabe \n ",i);

                   break;



                  }

          }

       printf("\n the available now is %d :  ",i);
```

```
        for(j=0;j<num_resourses;j++)      {

            if(check_available==true){
                  check[i]=1;  // to don't repeat

                    available[i+1][j]=available[i][j]+allocation[i][j];
                    printf("   %d",available[i+1][j]);

            }


        else{
            available[i+1][j]=available[i][j];
              printf(" %d",available[i][j]);
        }

  }


    if(check[i]==1)
    printf("\n\n process P %d done \n",i);

    finish--;

  }

  }

          return EXIT_SUCCESS;
  }
```