NAME-AHANA BISWAS ---ID-201001001074---BATCH-BCS3A---DATE-02.12.2022---GROUP-B

# TECHNO INDIA UNIVERSITY, WEST BENGAL

## STREAM-B.Tech CSE

## OPERATING SYSTEMS LAB SUBJECT CODE:-TIU-UCS-T30#

## ASSIGNMENT-05 SEMESTER-5

## Write the code in C language and show the output MARKS:20

**1) Do the following simulation to check the safe state of the system at a specific time stamp— (a) Single Resource, Multiple Instances: Consider there are n processes and one resource having multiple instances. The allocation vector A[ ] represents the current allocation of the resource instances concerned to the corresponding process. Max vector M[ ] represents the maximum requirement of the resource instances concerned to the corresponding process. The total number of instances of the resource and currently available resource instances are stored in the Totinst and Curinst respectively. User will give the inputs as follows:i) number of processes ii) Totinst and Curinst iii) allocation vector A[ ] iv) Max vector M[ ]. We need to check the system is safe or not. If safe find all possible safe sequences.**

**(b) Multiple Resource, Multiple Instances(Bankers safety algorithm): Consider there are n processes and every resource having multiple instances. The allocation Matrix A[ ][ ] represents the current allocation of the resource instances concerned to the corresponding process. Max Matrix M[ ][ ] represents the maximum requirement of the resource instances concerned to the corresponding process. The total number of instances of the resources and currently available resource's instances are stored in the vector Totinst[ ] and Curinst[ ] respectively. User will give the inputs as follows:i) number of processes ii) Totinst[ ] and Curinst[ ] iii) allocation Matrix A[ ][ ] iv) Max Matrix M[ ][ ]. We need to check the system is safe or not. If safe find all possible safe sequences.**

**CODE (a) –**

```
#include<stdio.h>

int all[10],max[10],need[10],ss[10],totinst,curinst,n,work=0,d_a[10],k=0;
```

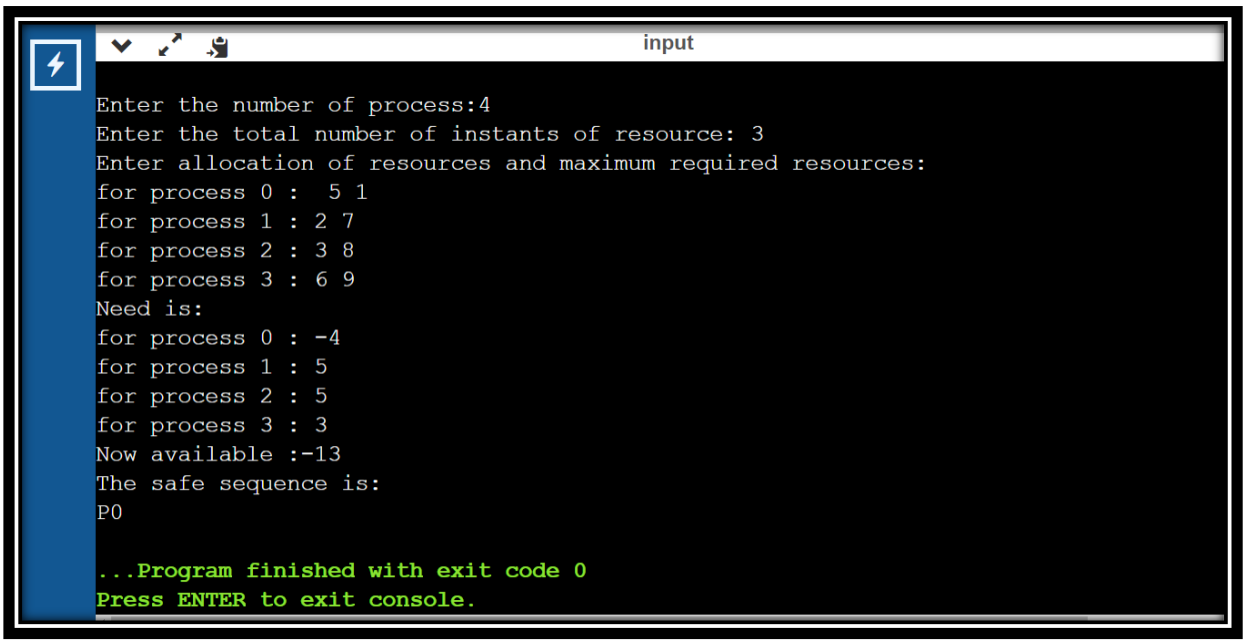
```
void findsafe();
int main()
{       int i,s=0;
        printf("\nEnter the number of process:");
        scanf("%d",&n);
        printf("Enter the total number of instants of resource:");
        scanf("%d",&totinst);
        printf("Enter allocation of resources and maximum required
resources:\n");
        for(i=0;i<n;i++)
        {       printf("for process %d : ",i);
                d_a[i]=0;
                scanf("%d",&all[i]);
                s=s+all[i];
                scanf("%d",&max[i]);
        }
        printf("Need is:\n");
        for(i=0;i<n;i++)
        {       printf("for process %d : ",i);
                need[i]=max[i]-all[i];
                printf("%d\n",need[i]);
        }
        curinst=totinst-s;
        printf("Now available :%d ",curinst);
```

```
        findsafe();
        printf("\nThe safe sequence is:\n");
        for(i=0;i<(k-1);i++)
        {       printf("P%d-->",ss[i]);}
        printf("P%d",ss[k-1]);
return 0;
}

void findsafe()
{               int work=curinst;
                int i,j;
                for(j=1;j<=n;j++)
                {       for(i=0;i<n;i++)
                        {       if(d_a[i]==0)
                                        {       if(need[i]<=work)
                                                {       ss[k]=i;
                                                        k++;
                                                        d_a[i]=1;
                                                        work=work+all[i];
                                                }
                                        }
                                else
```

```
                        {       continue;
                        }
                }
        }
}
```

**OUTPUT (a) –**




**CODE (b)–**

```
#include<stdio.h>

int
all[10][10],max[10][10],need[10][10],ss[10],totinst[10],curinst[10],n,m,work[10],d_a[10],k=0;

void findsafe();

int cheak(int );

void updatework(int );

int main()

{       int i,s[10],j;
```

```
printf("\nEnter the number of process:");
scanf("%d",&n);
printf("Enter the total number of resource:");
scanf("%d",&m);
printf("Enter total instances of all resources:\n");
for(i=0;i<m;i++)
{       printf("Enter total instance of %d resource:",i);
        scanf("%d",&totinst[i]);
}
printf("Enter allocation of instance of resources:\n");
for(i=0;i<n;i++)
{       printf("for process %d : ",i);
        d_a[i]=0;
        for(j=0;j<m;j++)
        {       s[j]=0;
                scanf("%d",&all[i][j]);}
}
printf("Enter maximum of instance of resources:\n");
for(i=0;i<n;i++)
{       printf("for process %d : ",i);
        for(j=0;j<m;j++)
        {scanf("%d",&max[i][j]);}
}
```

```
printf("Need is:\n");
for(i=0;i<n;i++)
{       printf("\nfor process %d : ",i);
        for(j=0;j<m;j++)
        {need[i][j]=max[i][j]-all[i][j];
        printf("%d ",need[i][j]);}
}
j=0;
do{
for(i=0;i<n;i++)
{       s[j]=s[j]+all[i][j];}
j++;
}while(j<m);
printf("\nNow available :\n");
for(i=0;i<m;i++)
{       curinst[i]=totinst[i]-s[i];
printf("%d ",curinst[i]);}

findsafe();
printf("\nThe safe sequence is:\n");
for(i=0;i<(k-1);i++)
{       printf("P%d-->",ss[i]);}
printf("P%d",ss[k-1]);
```

```
return 0;
}

void findsafe()
{
        int i,j,h;
        for(i=0;i<m;i++)
        {       work[i]=curinst[i];}
        for(j=1;j<=n;j++)
        {       for(i=0;i<n;i++)
                {       if(d_a[i]==0)
                                {       h=cheak(i);
                                        if(h==1)
                                        {       ss[k]=i;
                                                k++;
                                                d_a[i]=1;
                                                updatework(i);
                                        }
                                }
                        else
                        {       continue;
                        }
                }
```

```
                }
}

int cheak(int l)
{       int z=0,p=0;
        for(p=0;p<m;p++)
        {       if(need[l][p]<=work[p])
                        {       z=1;}
                else
                        {       z=0;
                                break;}
        }
return(z);
}

void updatework(int l)
{       int p=0;
        for(p=0;p<m;p++)
        {       work[p]=work[p]+all[l][p];
        }
}
```

NAME-AHANA BISWAS ---ID-201001001074---BATCH-BCS3A---DATE-02.12.2022---GROUP-B

## OUTPUT (b) –