OPERATING SYSTEMS LAB

BANKER’S ALGORITHM

SREYAS V

CSE C

185001162

#include<stdio.h>

#include<string.h>

#include<math.h>

typedef struct

{

int alloc[10];

int max[10];

int need[10];

int comp;

}process;

int safe(process p[], int avail[], int nop, int nor)

{

int i,j,k,flag=1,flag2;

int save[50];

k=0;

i=0;

while(flag==1)

{

flag=0;

for(;i<nop;i++)

{

flag2=0;

for(j=0;j<nor;j++)

{

if(p[i].need[j]>avail[j] || p[i].comp!=0)

flag2=1;

}

if(flag2==0)

{

p[i].comp=1;

flag=1;

for(j=0;j<nor;j++)

{

avail[j]+=p[i].alloc[j];

}

save[k]=i;

k++;

}

}

if(i==nop)

i=0;

}

if(k<nop-1)

{

printf("Unsafe State\n");

return 0;

}

else

{

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

printf("%d ",save[j]);

printf("\n");

return 1;

}

}

void display(process p[], int avail[], int nop, int nor)

{

int i,j,k;

printf("Process\t\t");

printf("Alloc\t");

for(i=1;i<nor;i++) printf("\t");

printf("Need\t");

for(i=1;i<nor;i++) printf("\t");

printf("Avail\n");

printf("\t\t");

for(i=0;i<nor;i++) printf("%d\t",i);

for(i=0;i<nor;i++) printf("%d\t",i);

for(i=0;i<nor;i++) printf("%d\t",i);

printf("\n\n");

for(i=0;i<nop;i++)

{

printf("%d\t\t",i);

for(j=0;j<nor;j++) printf("%d\t",p[i].alloc[j]);

for(j=0;j<nor;j++) printf("%d\t",p[i].need[j]);

if(i==0)

for(j=0;j<nor;j++) printf("%d\t",avail[j]);

printf("\n");

}

}

int main()

{

int nop;

int nor;

int avail[10];

process p2[50];

int avail2[10];

int choice;

process p[50];

int r;

int i,j,k,l;

int pro;

do

{

printf("1.Read Data\n2.Print Data\n3.Check State\n4.Process Request\n5.Exit\nEnter Choice:");

scanf("%d",&choice);

switch(choice)

{

case 1:

printf("Enter no of resources: ");

scanf("%d",&nor);

for(i=0;i<nor;i++)

{

printf("Enter no of instances of resource available %d: ",i);

scanf("%d",&avail[i]);

}

printf("Enter no of processes: ");

scanf("%d",&nop);

for(i=0;i<nop;i++)

{

printf("Data for process %d: \n",i);

printf("Max: \n");

for(j=0;j<nor;j++)

{

printf("Enter max no of instances of resource %d required: ",j);

scanf("%d",&p[i].max[j]);

}

printf("Allocation: \n");

for(j=0;j<nor;j++)

{

printf("Enter no of instances of resource %d allocated : ",j);

scanf("%d",&p[i].alloc[j]);

}

for(j=0;j<nor;j++)

{

p[i].need[j]=p[i].max[j]-p[i].alloc[j];

}

p[i].comp=0;

}

break;

case 2:

display(p,avail,nop,nor);

break;

case 3:

for(i=0;i<nop;i++)

{

p2[i]=p[i];

}

for(i=0;i<nor;i++)

{

avail2[i]=avail[i];

}

safe(p2,avail2,nop,nor);

break;

case 4:

printf("Enter the process no: ");

scanf("%d",&pro);

for(i=0;i<nop;i++)

{

p2[i]=p[i];

}

for(i=0;i<nor;i++)

{

avail2[i]=avail[i];

}

for(j=0;j<nor;j++)

{

printf("Enter the request for resource %d: ",j);

scanf("%d",&r);

if(r<=avail2[j] && r<=p2[pro].need[j])

{

p2[pro].need[j]-=r;

p2[pro].alloc[j]+=r;

avail2[j]-=r;

}

}

if(safe(p2,avail2,nop,nor))

printf("Safe state found. Process can be allocated with the necessary resources.\n");

else

printf("Allocation causes unsafe state. Process request cannot be granted.\n");

break;

}

}while(choice!=5);

}

OUTPUT

/\*

1.Read Data

2.Print Data

3.Check State

4.Process Request

5.Exit

Enter Choice:1

Enter no of resources: 4

Enter no of instances of resource available 0: 3

Enter no of instances of resource available 1: 3

Enter no of instances of resource available 2: 2

Enter no of instances of resource available 3: 1

Enter no of processes: 5

Data for process 0:

Max:

Enter max no of instances of resource 0 required: 4

Enter max no of instances of resource 1 required: 2

Enter max no of instances of resource 2 required: 1

Enter max no of instances of resource 3 required: 2

Allocation:

Enter no of instances of resource 0 allocated : 2

Enter no of instances of resource 1 allocated : 0

Enter no of instances of resource 2 allocated : 0

Enter no of instances of resource 3 allocated : 1

Data for process 1:

Max:

Enter max no of instances of resource 0 required: 5

Enter max no of instances of resource 1 required: 2

Enter max no of instances of resource 2 required: 5

Enter max no of instances of resource 3 required: 2

Allocation:

Enter no of instances of resource 0 allocated : 3

Enter no of instances of resource 1 allocated : 1

Enter no of instances of resource 2 allocated : 2

Enter no of instances of resource 3 allocated : 1

Data for process 2:

Max:

Enter max no of instances of resource 0 required: 2

Enter max no of instances of resource 1 required: 3

Enter max no of instances of resource 2 required: 1

Enter max no of instances of resource 3 required: 6

Allocation:

Enter no of instances of resource 0 allocated : 2

Enter no of instances of resource 1 allocated : 1

Enter no of instances of resource 2 allocated : 0

Enter no of instances of resource 3 allocated : 3

Data for process 3:

Max:

Enter max no of instances of resource 0 required: 1

Enter max no of instances of resource 1 required: 4

Enter max no of instances of resource 2 required: 2

Enter max no of instances of resource 3 required: 4

Allocation:

Enter no of instances of resource 0 allocated : 1

Enter no of instances of resource 1 allocated : 3

Enter no of instances of resource 2 allocated : 1

Enter no of instances of resource 3 allocated : 2

Data for process 4:

Max:

Enter max no of instances of resource 0 required: 3

Enter max no of instances of resource 1 required: 6

Enter max no of instances of resource 2 required: 6

Enter max no of instances of resource 3 required: 5

Allocation:

Enter no of instances of resource 0 allocated : 1

Enter no of instances of resource 1 allocated : 4

Enter no of instances of resource 2 allocated : 3

Enter no of instances of resource 3 allocated : 2

1.Read Data

2.Print Data

3.Check State

4.Process Request

5.Exit

Enter Choice:2

Process Alloc Need Avail

0 1 2 3 0 1 2 3 0 1 2 3

0 2 0 0 1 2 2 1 1 3 3 2 1

1 3 1 2 1 2 1 3 1

2 2 1 0 3 0 2 1 3

3 1 3 1 2 0 1 1 2

4 1 4 3 2 2 2 3 3

1.Read Data

2.Print Data

3.Check State

4.Process Request

5.Exit

Enter Choice:3

0 3 4 1 2

1.Read Data

2.Print Data

3.Check State

4.Process Request

5.Exit

Enter Choice:4

Enter the process no: 1

Enter the request for resource 0: 1

Enter the request for resource 1: 1

Enter the request for resource 2: 0

Enter the request for resource 3: 0

0 3 4 1 2

Safe state found. Process can be allocated with the necessary resources.

1.Read Data

2.Print Data

3.Check State

4.Process Request

5.Exit

Enter Choice:5\*/