PAGING TECHNIQUES

kaushik@kaushik-AcerPower-Series:~/OS $ cat pagingtech.c

```
#include<stdio.h>
#include<stdlib.h>
#include<malloc.h>
typedef struct Pnode* Process;
typedef struct Fnode* FreeFrame;
struct Pnode
{
char ID[2];
int number,page[10];
Process Next;
};
struct Fnode
{
int Number;
FreeFrame Next;
};
Process PCreate()
{
Process head;
head=(struct Pnode*)malloc(sizeof(struct Pnode));
head->Next=NULL;
return head;
}
FreeFrame FCreate()
{
FreeFrame head;
head=(struct Fnode*)malloc(sizeof(struct Fnode));
head->Next=NULL;
return head;
}
void FrameAllocation(int Number,FreeFrame F)
{
FreeFrame pos,NewFrame;
pos=F;
while(pos->Next!=NULL)
pos=pos->Next;
NewFrame=(struct Fnode*)malloc(sizeof(struct Fnode));
NewFrame->Number=Number;
NewFrame->Next=NULL;
pos->Next=NewFrame;
return;
}
void DisplayFrame(FreeFrame F)
{
printf("\nThe Free Frames are: ");
FreeFrame pos;
pos=F->Next;
while(pos!=NULL)
{
printf(" %d ",pos->Number);
pos=pos->Next;
}
printf("\n");
return;
}
void DisplayPageTable(Process P)
{
int i;
Process pos;
pos=P->Next;
while(pos!=NULL)
```

```
{
printf("\nPAGE TABLE FOR PROCESS %s",pos->ID);
for(i=0;i<pos->number;i++)
printf("\nPage %d : Frame %d",i,pos->page[i]);
pos=pos->Next;
printf("\n");
} return;
}
int ProcessRequest(Process P,FreeFrame F,int PageSize,int numberFreeFrames)
{
int i,size;
Process pos,NewProcess;
FreeFrame temp;
NewProcess=(struct Pnode*)malloc(sizeof(struct Pnode));
printf("Enter the Process Requirement(ID and Size): ");
scanf("%s%d",NewProcess->ID,&size);
NewProcess->number=size/PageSize;
if(NewProcess->number>numberFreeFrames)
{
printf("\nThe Process Requirement is not available\n");
free(NewProcess);
return numberFreeFrames;
}
numberFreeFrames-=NewProcess->number;
printf("The Process is divided into %d Pages\n",NewProcess->number);
for(i=0;i<NewProcess->number;i++)
{
temp=F->Next;
NewProcess->page[i]=temp->Number;
F->Next=temp->Next;
free(temp);
}
NewProcess->Next=NULL;
pos=P;
while(pos->Next!=NULL)
pos=pos->Next;
pos->Next=NewProcess;
DisplayPageTable(pos);
return numberFreeFrames;
}
void ProcessDealloc(Process P,FreeFrame F,char ID[])
{
int i;
Process pos,temp;
pos=P;
while(pos->Next!=NULL&&strcmp(pos->Next->ID,ID)!=0)
pos=pos->Next;
temp=pos->Next;
for(i=0;i<temp->number;i++)
FrameAllocation(temp->page[i],F);
pos->Next=temp->Next;
free(temp);
printf("\nThe Process %s has been deallocated\n",ID);
return;
}
void main()
{
Process P;
FreeFrame F;
F=FCreate();
P=PCreate();
int i,memSize,pageSize,numberFrames,numberFreeFrames=10,choice=-1;
```

```
char ID[2];
printf("\nPAGING TECHNIQUE\n");
printf("Enter the Physical Memory Size(in Kb): ");
scanf("%d",&memSize);
printf("Enter the Page Size(in Kb): ");
scanf("%d",&pageSize);
numberFrames=memSize/pageSize;
printf("\nPhysical Memory is divided in %d Frames",numberFrames);
printf("\nAfter Initialisation...");
for(i=0;i<numberFreeFrames;i++)
FrameAllocation(rand()%numberFrames,F);
DisplayFrame(F);
printf("\n\n1. PROCESS REQUEST\n2. DEALLOCATION\n3. PAGE TABLE DISPLAY FOR ALL INPUT
PROCESSES\n4. FREE FRAME LIST DISPLAY\n5. EXIT\n");
while(choice!=5)
{
printf("\nEnter an option: ");
scanf("%d",&choice);
switch(choice)
{
case 1://PROCESS REQUEST
numberFreeFrames=ProcessRequest(P,F,pageSize,numberFreeFrames);
break;
case 2://DEALLOCATION
printf("Enter the Process ID to be deallocated: ");
scanf("%s",ID);
ProcessDealloc(P,F,ID);
break;
case 3://PAGE TABLE DISPLAY FOR ALL INPUT PROCESSES
DisplayPageTable(P);
break;
case 4://FREE FRAME LIST DISPLAY
DisplayFrame(F);
break;
case 5://EXIT
break;
}
}
}
kaushik@kaushik-AcerPower-Series:~/OS $
PAGING TECHNIQUE
Enter the Physical Memory Size(in Kb): 64
Enter the Page Size(in Kb): 2
Physical Memory is divided in 32 Frames
After Initialisation...
The Free Frames are: 7 6 9 19 17 31 10 12 9 13
1. PROCESS REQUEST
2. DEALLOCATION
3. PAGE TABLE DISPLAY FOR ALL INPUT PROCESSES
4. FREE FRAME LIST DISPLAY
5. EXIT
Enter an option: 1
Enter the Process Requirement(ID and Size): P1 6
The Process is divided into 3 Pages
PAGE TABLE FOR PROCESS P1
Page 0 : Frame 7
Page 1 : Frame 6
Page 2 : Frame 9
Enter an option: 4
The Free Frames are: 19 17 31 10 12 9 13
Enter an option: 1
Enter the Process Requirement(ID and Size): P2 8
```

```
The Process is divided into 4 Pages
PAGE TABLE FOR PROCESS P2
Page 0 : Frame 19
Page 1 : Frame 17
Page 2 : Frame 31
Page 3 : Frame 10
Enter an option: 4
The Free Frames are: 12 9 13
Enter an option: 1
Enter the Process Requirement(ID and Size): P3 8
The Process Requirement is not available
Enter an option: 4
The Free Frames are: 12 9 13
Enter an option: 3
PAGE TABLE FOR PROCESS P1
Page 0 : Frame 7
Page 1 : Frame 6
Page 2 : Frame 9
PAGE TABLE FOR PROCESS P2
Page 0 : Frame 19
Page 1 : Frame 17
Page 2 : Frame 31
Page 3 : Frame 10
Enter an option: 2
Enter the Process ID to be deallocated: P1
The Process P1 has been deallocated
Enter an option: 4
The Free Frames are: 12 9 13 7 6 9
Enter an option: 3
PAGE TABLE FOR PROCESS P2
Page 0 : Frame 19
Page 1 : Frame 17
Page 2 : Frame 31
Page 3 : Frame 10
Enter an option: 5

kaushik@kaushik-AcerPower-Series:~/OS $
kaushik@kaushik-AcerPower-Series:~/OS $
```