21BCE7371
RADHA KRISHNA GARG

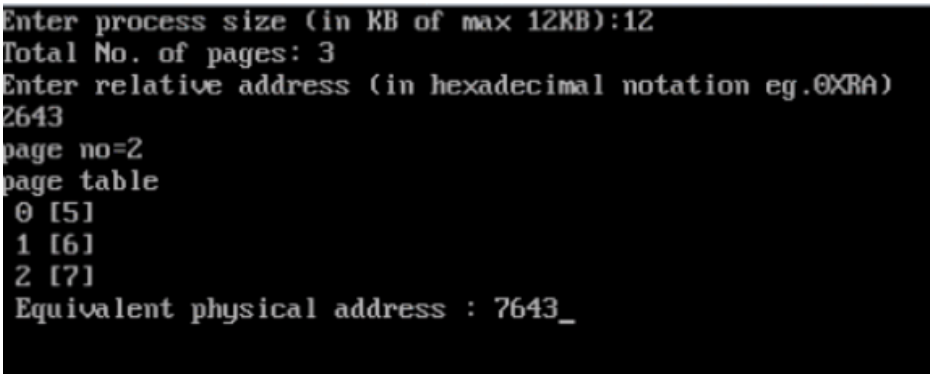
OS LAB ASSIGNMENT
MEMORY MANAGEMENT

1.Implementation of memory management using paging.

Using PAGING

Paging Technique

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
int size,m,n,pgno,pagetable[3]={5,6,7},i,j,frameno;
double m1;
int ra=0,ofs;
clrscr();
printf("Enter process size (in KB of max 12KB):");/*reading memeory size*/
scanf("%d",&size);
m1=size/4;
n=ceil(m1);
printf("Total No. of pages: %d",n);
printf("\nEnter relative address (in hexadecimal notation eg.0XRA) \n");
//printf("The length of relative Address is : 16 bits \n\n The size of offset is :12 bits\n");
scanf("%d",&ra);
pgno=ra/1000; /*calculating physical address*/
ofs=ra%1000;
printf("page no=%d\n",pgno);
printf("page table");
for(i=0;i<n;i++)
printf("\n %d [%d]",i,pagetable[i]);
frameno=pagetable[pgno];
printf("\n Equivalent physical address : %d%d",frameno,ofs);
getch();
}
```



2.Implementation of memory management using segmentation.

Using SEGMENTATION

SOURCE CODE:

```
#include<stdio.h>
#include<conio.h>
struct list
{
int seg;
int base;
int limit;
struct list *next;
} *p;
void insert(struct list *q,int base,int limit,int seg)
{
if(p==NULL)
{
p=malloc(sizeof(Struct list));
p->limit=limit;
p->base=base;
p->seg=seg;
p->next=NULL;
}
else
{
while(q->next!=NULL)
{
Q=q->next;
Printf(“yes”)
}
q->next=malloc(sizeof(Struct list));
q->next ->limit=limit;
q->next ->base=base;
q->next ->seg=seg;
```

```
q->next ->next=NULL;
}
}
int find(struct list *q,int seg)
{
while(q->seg!=seg)
{
q=q->next;
}
return q->limit;
}
int search(struct list *q,int seg)
{
while(q->seg!=seg)
{
q=q->next;
}
return q->base;
}
main()
{
p=NULL;
int seg,offset,limit,base,c,s,physical;
printf(“Enter segment table/n”);
printf(“Enter -1 as segment value for termination\n”);
do
{
printf(“Enter segment number”);
scanf(“%d”,&seg);
if(seg!=-1)
{
printf(“Enter base value:”);
scanf(“%d”,&base);
printf(“Enter value for limit:”);
scanf(“%d”,&limit);
insert(p,base,lmit,seg);
}
}
while(seg!=-1)
printf(“Enter offset:”);
scanf(“%d”,&offset);
printf(“Enter bsegmentation number:”);
scanf(“%d”,&seg);
c=find(p,seg);
s=search(p,seg);
if(offset<c)
{
physical=s+offset;
```

```
printf(“Address in physical memory %d\n”,physical);
}
else
{
printf(“error”);
}
```

OUTPUT:

```
 $ cc seg.c
 $ ./a.out
Enter segment table
Enter -1 as segmentation value for termination
Enter segment number:1
Enter base value:2000
Enter value for limit:100
Enter segment number:2
Enter base value:2500
Enter value for limit:100
Enter segmentation number:-1
Enter offset:90
Enter segment number:2
Address in physical memory 2590
$ ./a.out
Enter segment table
Enter -1 as segmentation value for termination
Enter segment number:1
Enter base value:2000
Enter value for limit:100
Enter segment number:2
Enter base value:2500
Enter value for limit:100
Enter segmentation number:-1
Enter offset:90
Enter segment number:1
Address in physical memory 2090
}
```