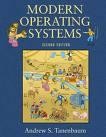
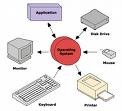


PAGING-MEMORY ALLOCATION TECHNIQUE

**LAB MANUAL 14**



|  |
| --- |
| **Date:** |
| **Name:** |
| **Reg#: Group:** |
| **Marks: Signature:** |



**Operating systems**

**Lab objective**

To implement the Memory management policy- Paging.

**Algorithm**

Step 1: Read all the necessary input from the keyboard.

Step 2: Pages - Logical memory is broken into fixed - sized blocks. Step 3: Frames – Physical memory is broken into fixed – sized blocks. Step 4: Calculate the physical address using the following

Physical address = ( Frame number \* Frame size ) + offset Step 5: Display the physical address.

Step 6: Stop the process.

**/\* Memory Allocation with Paging Technique \*/**

#include <stdio.h> #include <conio.h> struct pstruct

{

int fno; int pbit;

}ptable[10];

int pmsize,lmsize,psize,frame,page,ftable[20],frameno; void info()

{

printf("\n\nMEMORY MANAGEMENT USING PAGING\n\n"); printf("\n\nEnter the Size of Physical memory: "); scanf("%d",&pmsize);

printf("\n\nEnter the size of Logical memory: "); scanf("%d",&lmsize);

printf("\n\nEnter the partition size: "); scanf("%d",&psize);

frame = (int) pmsize/psize; page = (int) lmsize/psize;

printf("\nThe physical memory is divided into %d no.of frames\n",frame); printf("\nThe Logical memory is divided into %d no.of pages",page);

}

void assign()



**Operating systems**

{

int i;

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

{

ptable[i].fno = -1;

ptable[i].pbit= -1;

}

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

ftable[i] = 32555; for (i=0;i<page;i++)

{

printf("\n\nEnter the Frame number where page %d must be placed: ",i); scanf("%d",&frameno);

ftable[frameno] = i; if(ptable[i].pbit == -1)

{

ptable[i].fno = frameno; ptable[i].pbit = 1;

}

}

getch();

// clrscr();

printf("\n\nPAGE TABLE\n\n"); printf("PageAddress FrameNo. PresenceBit\n\n"); for (i=0;i<page;i++)

printf("%d\t\t%d\t\t%d\n",i,ptable[i].fno,ptable[i].pbit); printf("\n\n\n\tFRAME TABLE\n\n");

printf("FrameAddress PageNo\n\n"); for(i=0;i<frame;i++)

printf("%d\t\t%d\n",i,ftable[i]);

}

void cphyaddr()

{

int laddr,paddr,disp,phyaddr,baddr; getch();

// clrscr();

printf("\n\n\n\tProcess to create the Physical Address\n\n"); printf("\nEnter the Base Address: ");

scanf("%d",&baddr);

printf("\nEnter theLogical Address: "); scanf("%d",&laddr);

paddr = laddr / psize; disp = laddr % psize;

if(ptable[paddr].pbit == 1 )

phyaddr = baddr + (ptable[paddr].fno\*psize) + disp;



**Operating systems**

printf("\nThe Physical Address where the instruction present: %d",phyaddr);

}

void main()

{

clrscr();

info();

assign(); cphyaddr(); getch();

}

**OUTPUT:**

**MEMORY MANAGEMENT USING PAGING**

Enter the Size of Physical memory: 16 Enter the size of Logical memory: 8 Enter the partition size: 2

The physical memory is divided into 8 no.of frames The Logical memory is divided into 4 no.of pages

Enter the Frame number where page 0 must be placed: 5 Enter the Frame number where page 1 must be placed: 6 Enter the Frame number where page 2 must be placed: 7 Enter the Frame number where page 3 must be placed: 2

|  |  |  |
| --- | --- | --- |
| **PAGE TABLE** |  | |
| **PageAddress** | **FrameNo.** | **PresenceBit** |
| 0 | 5 | 1 |
| 1 | 6 | 1 |
| 2 | 7 | 1 |
| 3 | 2 | 1 |
| **FRAME TABLE**  **FrameAddress** | **PageNo** |  |
| 0 | 32555 |  |
| 1 | 32555 |  |
| 2 | 3 |  |
| 3 | 32555 |  |



**Operating systems**

4 32555

5 0

6 1

7 2

**Process to create the Physical Address**

Enter the Base Address: 1000 Enter theLogical Address: 3

The Physical Address where the instruction present: 1013