**Practical : 14**

**Aim: Implementation of Optimal Page Replacement Algorithm.**

**Program:**

#include<stdio.h>

int main()

{

int frame\_size, no\_of\_pages, flag1, flag2, flag3, i, j, k, pos, max, page\_faults = 0, hit = 0;

int pri[20][30], frames[10], pages[30], temp[10];

char x[20];

printf("Enter number of pages: ");

scanf("%d", &no\_of\_pages);

printf("Enter the pages: ");

for(i = 0; i < no\_of\_pages; ++i)

{

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

}

printf("Enter number of frames: ");

scanf("%d", &frame\_size);

for(i=0; i< no\_of\_pages; i++)

{

x[i]='X';

}

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

{

frames[i] = -1;

}

for(i = 0; i < no\_of\_pages; ++i)

{

flag1 = flag2 = 0;

for(j = 0; j < frame\_size; ++j)

{

if(frames[j] == pages[i])

{

flag1 = flag2 = 1;

x[i]='\*';

break;

}

}

if(flag1 == 0)

{

for(j = 0; j < frame\_size; ++j)

{

if(frames[j] == -1)

{

page\_faults++;

//x[j]='\*';

frames[j] = pages[i];

flag2 = 1;

break;

}

}

}

if(flag2 == 0)

{

flag3 =0;

for(j = 0; j < frame\_size; ++j)

{

temp[j] = -1;

for(k = i + 1; k < no\_of\_pages; ++k)

{

if(frames[j] == pages[k])

{

temp[j] = k;

break;

}

}

}

for(j = 0; j < frame\_size; ++j)

{

if(temp[j] == -1)

{

pos = j;

flag3 = 1;

break;

}

}

if(flag3 ==0)

{

max = temp[0];

pos = 0;

for(j = 1; j < frame\_size; ++j)

{

if(temp[j] > max)

{

max = temp[j];

pos = j;

}

}

}

frames[pos] = pages[i];

page\_faults++;

}

for(j = 0; j < frame\_size; j++)

{

pri[j][i]=frames[j];

}

}

for(i = 0 ; i < no\_of\_pages\*5+2\*frame\_size+1; i ++)

{

printf("-");

}

// printf("\n|Frames |");

printf("\n| |");

for(i=0; i<= 2\* no\_of\_pages + 6; i++)

{

printf(" ");

}

printf("Pages");

for(i=0; i<= 2\*no\_of\_pages + 6; i++)

{

printf(" ");

}

printf("|");

printf("\n|Frames |");

for(i = 0 ; i < no\_of\_pages\*5+2\*frame\_size - 8; i ++)

{

printf("-");

}

printf("\n|\t|");

for(i=0 ; i < no\_of\_pages ; i++)

{

printf(" %2d |",pages[i]);

}

printf("\n|");

for(i = 0 ; i < no\_of\_pages\*5+2\*frame\_size + 0; i ++)

{

printf("-");

}

printf("\n");

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

{

printf("| %2d",i);

printf("\t|");

for(j = 0; j < no\_of\_pages; j++)

{

if(pri[i][j]== -1)

{

printf(" - |");

}

else

printf(" %2d |",pri[i][j]);

}

printf("\n");

}

for(i = 0 ; i < no\_of\_pages\*5+2\*frame\_size+1; i ++)

{

printf("-");

}

printf("\n| |");

for(i = 0; i< no\_of\_pages; i++)

{

if(x[i]=='X')

{

printf("\033[0;31m");

printf(" %2c ",x[i]);

printf("\033[0m");

printf("|");

}

else

{

printf("\033[0;32m");

printf(" %2c ",x[i]);

printf("\033[0m");

printf("|");

}

}

//printf("\033[0m")

printf("\n");

for(i = 0 ; i < no\_of\_pages\*5+2\*frame\_size+1; i ++)

{

printf("-");

}

hit = no\_of\_pages-page\_faults;

printf("\n\n Total Page Faults = Total No of pages - Total Pages hits \n");

printf(" = %d - %d \n",no\_of\_pages,hit);

printf(" = %d \n",page\_faults);

printf("\n Total Page Hits = Total No of pages - Total Pages Miss \n ");

printf(" = %d - %d \n",no\_of\_pages,page\_faults);

printf(" = %d \n",hit);

printf("\nTotal Page Fault ratio = Total Page faults / Total pages \n");

printf(" = %d / %d \n",page\_faults,no\_of\_pages);

printf(" = %5.2f \n",((float)page\_faults/no\_of\_pages));

printf("\nTotal Page Hit ratio = Total Page hits / Total pages \n");

printf(" = %d / %d \n",(no\_of\_pages-page\_faults),no\_of\_pages);

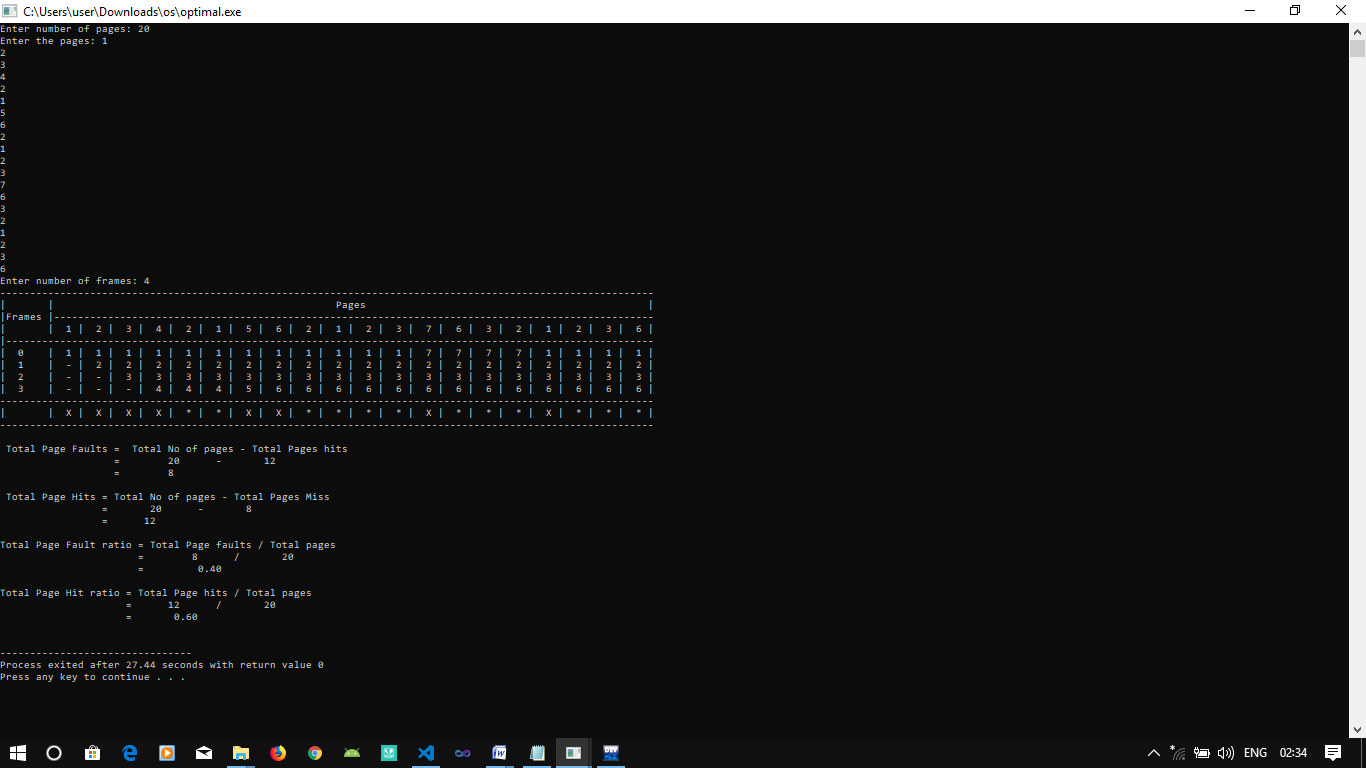
printf(" = %5.2f \n",((float)no\_of\_pages-page\_faults)/no\_of\_pages);

printf("\n");

return 0;

}

**Output :**

****