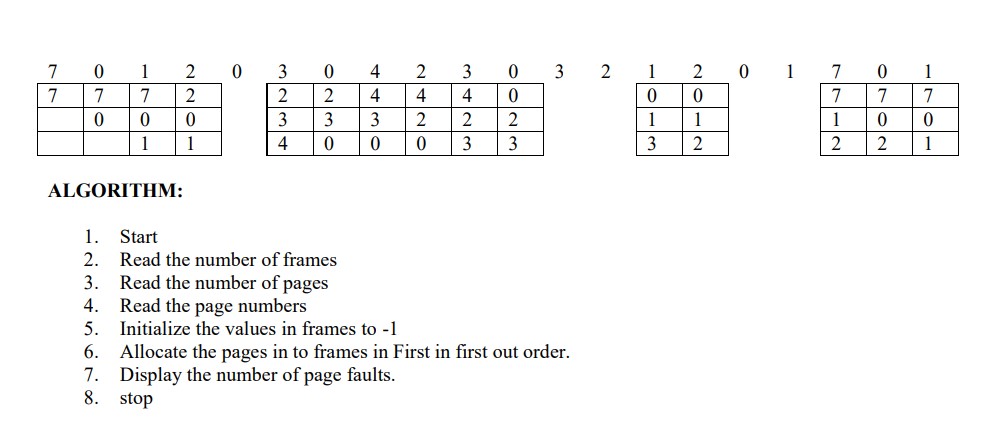
14. Simulate the following page replacement algorithms

a) FIFO b) LRU c) LFU etc.,

AIM: Simulate FIFO page replacement algorithms

FIFO algorithm: The simpler page replacement algorithm is a FIFO algorithm. A FIFO replacement algorithm associates with each page the time when that page was brought into memory. When a page must be replace, the oldest page is chosen. We can create a FIFO queue to hold all pages in memory. We replace the page at the head of the queue when a page is brought into memory; we insert it at the tail of the queue



PROGRAM

FIFO PAGE REPLACEMENT ALGORITHM

#include<stdio.h>

#include<conio.h>

void main()

{

int i, j, k, f, pf=0, count=0, rs[25], m[10], n;

printf("\n Enter the length of reference string --");

scanf("%d",&n);

printf("\n Enter the reference string --"); for(i=0;i<n;i++)

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

printf("\n Enter no. of frames --");

scanf("%d",&f);

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

m[i]=-1;

printf("\n The Page Replacement Process is --\n");

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

{

for(k=0;k<f;k++)

{

if(m[k]==rs[i])

break;

}

if(k==f)

{

m[count++]=rs[i];

pf++;

}

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

printf("\t%d",m[j]);

if(k==f)

printf("\tPF No. %d",pf); printf("\n");

if(count==f)

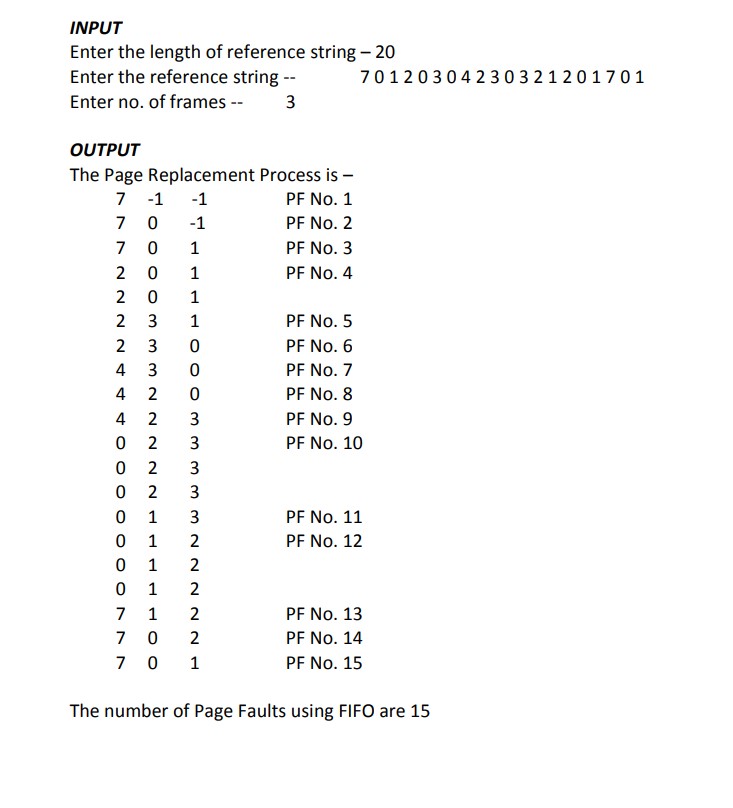
count=0;

}

printf("\n The number of Page Faults using FIFO are %d",pf);

getch();

}



b) LRU

AIM: Simulate LRU page replacement algorithms

1. Start

2. Read the number of frames

3. Read the number of pages

4. Read the page numbers

5. Initialize the values in frames to -1

6. Allocate the pages in to frames by selecting the page that has not been used for the longest period of time

7. Display the number of page faults.

8. stop

PROGRAM:

#include<stdio.h>

#include<conio.h>

void main()

{

int i, j , k, min, rs[25], m[10], count[10], flag[25], n, f, pf=0, next=1;

printf("Enter the length of reference string -- ");

scanf("%d",&n);

printf("Enter the reference string -- ");

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

{

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

flag[i]=0;

}

printf("Enter the number of frames -- ");

scanf("%d",&f);

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

{

count[i]=0; m[i]=-1;

}

printf("\nThe Page Replacement process is -- \n");

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

{

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

{

if(m[j]==rs[i])

{

flag[i]=1;

count[j]=next;

next++;

}

}

if(flag[i]==0)

{

if(i<f)

{

m[i]=rs[i];

count[i]=next;

next++;

}

else

{

min=0;

for(j=1;j<f;j++)

if(count[min] > count[j]) min=j;

m[min]=rs[i];

count[min]=next;

next++;

}

pf++;

}

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

printf("%d\t", m[j]);

if(flag[i]==0)

printf("PF No. -- %d" , pf);

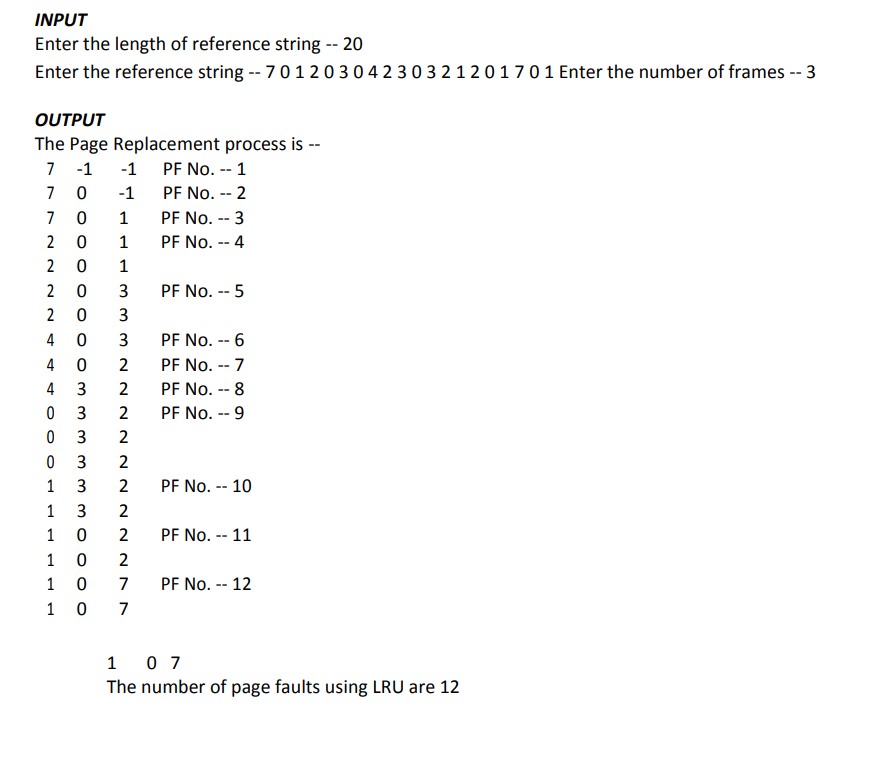
printf("\n");

}

printf("\nThe number of page faults using LRU are %d",pf);

getch();

}



c) LFU

AIM: Simulate LFU page replacement algorithms

ALGORITHM:

1. Start

2. Read the number of frames

3. Read the number of pages

4. Read the page numbers

5. Initialize the values in frames to -1

6. Allocate the pages in to frames by selecting the page that will not be used for the longest period of time.

7. Display the number of page faults.

8. stop

PROGRAM:

#include<stdio.h>

#include<conio.h>

void main()

{

int rs[50], i, j, k, m, f, cntr[20], a[20], min, pf=0;

printf("\nEnter number of page references -- ");

scanf("%d",&m);

printf("\nEnter the reference string -- ");

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

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

printf("\nEnter the available no. of frames -- ");

scanf("%d",&f);

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

{

cntr[i]=0;

a[i]=-1;

}

printf("\n The Page Replacement Process is –- \n");

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

{

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

if(rs[i]==a[j])

{

cntr[j]++;

break;

}

if(j==f)

{

min = 0;

for(k=1;k<f;k++)

if(cntr[k]<cntr[min])min=k;

a[min]=rs[i];

cntr[min]=1;

pf++;

}

printf("\n");

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

printf("\t%d",a[j]);

if(j==f)

printf("\tPF No. %d",pf);

}

printf("\n\n Total number of page faults -- %d",pf);

getch();

}

