Set A 1. Implementing FIFO

```
#include<stdio.h>
#define MAX 20
int frames[MAX],ref[MAX],mem[MAX][MAX],faults,sp=0,m,n;
void accept()
{
        int i;
        printf("Enter no.of frames:");
        scanf("%d", &n);
        printf("Enter no.of references:");
        scanf("%d", &m);
        printf("Enter reference string:\n");
        for(i=0;i<m;i++)
        {
                printf("[%d]=",i);
                scanf("%d",&ref[i]);
        }
}

void disp()
{
        int i,j;
        for(i=0;i<m;i++)
        printf("%3d",ref[i]);
        printf("\n\n");
        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        if(mem[i][j])
                        printf("%3d",mem[i][j]);
                        else
                                printf(" ");

                }
                printf("\n");
        }
        printf("Total Page Faults: %d\n",faults);
}

int search(int pno)
{
        int i;
        for(i=0;i<n;i++)
        {
                if(frames[i]==pno)
                return i;
        }
        return -1;
}
void fifo()
{
        int i,j;
        for(i=0;i<m;i++)
        {
                if(search(ref[i])==-1)
                {
                        frames[sp] = ref[i];
                        sp = (sp+1)%n;
                        faults++;
                        for(j=0;j<n;j++)
                        mem[j][i] = frames[j];
                }
        }
}

int main()
{
        accept();
        fifo();
        disp();
        return 0;
}
```

Output:

```
Enter no.of frames:3
Enter no.of references:12
Enter reference string:
[0]=1
[1]=2
[2]=3
[3]=4
[4]=1
[5]=2
[6]=5
[7]=1
[8]=2
[9]=3
[10]=4
[11]=5
1  2  3  4  1  2  5  1  2  3  4  5
```

```
  1  1  1  4  4  4  5     5  5
   2  2  2  1  1  1    3  3
    3  3  3  2  2    2  4
Total Page Faults: 9
```

2. Set A Implementing LRU

```
#include<stdio.h>
#define MAX 20

int frames[MAX],ref[MAX],mem[MAX][MAX],faults,
sp,m,n,time[MAX];
void accept()
{
        int i;
        printf("Enter no.of frames:");
        scanf("%d", &n);
        printf("Enter no.of references:");
        scanf("%d", &m);
        printf("Enter reference string:\n");
        for(i=0;i<m;i++)
        {
                printf("[%d]=",i);
                scanf("%d",&ref[i]);
        }
}

void disp()
        {
                int i,j;
                for(i=0;i<m;i++)
                printf("%3d",ref[i]);
                printf("\n\n");
                for(i=0;i<n;i++)
                {
                        for(j=0;j<m;j++)
                        {
                                if(mem[i][j])
                                printf("%3d",mem[i][j]);
                else
                printf(" ");
            }
            printf("\n");
        }
        printf("Total Page Faults: %d\n",faults);
    }

int search(int pno)
    {
        int i;
        for(i=0;i<n;i++)
        {
            if(frames[i]==pno)
            return i;
        }
        return -1;
    }
int get_lru()
    {
        int i,min_i,min=9999;
        for(i=0;i<n;i++)
        {
            if(time[i]<min)
            {
                min = time[i];
                min_i = i;
            }
        }
        return min_i;
    }

void lru()
    {
        int i,j,k;
        for(i=0;i<m && sp<n;i++)
        {
            k=search(ref[i]);
            if(k==-1)
            {
                frames[sp]=ref[i];
                time[sp]=i;
                faults++;
                sp++;
                for(j=0;j<n;j++)
                mem[j][i]=frames[j];
```

```
                        }
                        else
                        time[k]=i;
                }
                for(;i<m;i++)
                {
                        k = search(ref[i]);
                        if(k==-1)
                        {
                                sp = get_lru();
                                frames[sp] = ref[i];
                                time[sp] = i;
                                faults++;
                                for(j=0;j<n;j++)
                                mem[j][i] = frames[j];
                        }
                        else
                        time[k]=i;
                }
        }

int main()
{
        accept();
        lru();
        disp();
        return 0;
}
```

Output:

```
Enter no.of frames:3

Enter no.of references:12

Enter reference string:

[0]=1

[1]=2

[2]=3

[3]=4

[4]=1

[5]=2
```

```
[6]=5

[7]=1

[8]=2

[9]=3

[10]=4

[11]=5

1  2  3  4  1  2  5  1  2  3  4  5


  1  1  1  4  4  4  5    3  3  3

   2  2  2  1  1  1    1  4  4

    3  3  3  2  2    2  2  5

Total Page Faults: 1
```

3. Set B implementing OPT

```
#include<stdio.h>
int main()
{
   int no_of_frames, no_of_pages, frames[10], pages[30], temp[10], flag1, flag2, flag3, i, j, k, 
pos, max, faults = 0;
   printf("Enter number of frames: ");
   scanf("%d", &no_of_frames);

   printf("Enter number of pages: ");
   scanf("%d", &no_of_pages);

   printf("Enter page reference string: ");

   for(i = 0; i < no_of_pages; ++i){
      scanf("%d", &pages[i]);
   }
```

```
for(i = 0; i < no_of_frames; ++i){
    frames[i] = -1;
}

for(i = 0; i < no_of_pages; ++i){
    flag1 = flag2 = 0;

    for(j = 0; j < no_of_frames; ++j){
        if(frames[j] == pages[i]){
               flag1 = flag2 = 1;
               break;
           }
    }

    if(flag1 == 0){
        for(j = 0; j < no_of_frames; ++j){
            if(frames[j] == -1){
                faults++;
                frames[j] = pages[i];
                flag2 = 1;
                break;
            }
        }
    }

    if(flag2 == 0){
     flag3 =0;

        for(j = 0; j < no_of_frames; ++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 < no_of_frames; ++j){
         if(temp[j] == -1){
         pos = j;
         flag3 = 1;
         break;
         }
```

```
            }

            if(flag3 ==0){
             max = temp[0];
             pos = 0;

             for(j = 1; j < no_of_frames; ++j){
             if(temp[j] > max){
             max = temp[j];
             pos = j;
             }
             }
            }
frames[pos] = pages[i];
faults++;
        }

        printf("\n");

        for(j = 0; j < no_of_frames; ++j){
            printf("%d\t", frames[j]);
        }
    }

    printf("\n\nTotal Page Faults = %d", faults);

    return 0;
}
```

Output:

```
Enter number of frames: 3

Enter number of pages: 12

Enter page reference string: 1

2

3

4

1

2
```

```
5

1

2

3

4

5

1     -1     -1

1     2      -1

1     2      3

1     2      4

1     2      4

1     2      4

1     2      5

1     2      5

1     2      5

3     2      5

4     2      5

4     2      5


Total Page Faults = 7
```

4. Set B implementing MFU

```
#include<stdio.h>
#define MAX 20
```

```
int frames[MAX],ref[MAX],mem[MAX][MAX],faults,
 sp,m,n,count[MAX];

void accept()
        {
                int i;
                printf("Enter no.of frames:");
                scanf("%d", &n);

                printf("Enter no.of references:");
                scanf("%d", &m);

                printf("Enter reference string:\n");
                for(i=0;i<m;i++)
                {
                        printf("[%d]=",i);
                         scanf("%d",&ref[i]);
                }
        }

void disp()
{
         int i,j;

        for(i=0;i<m;i++)
         printf("%3d",ref[i]);
         printf("\n\n");
         for(i=0;i<n;i++)
         {
                 for(j=0;j<m;j++)
                 {
                         if(mem[i][j])
                        printf("%3d",mem[i][j]);
                else
                         printf("   ");
                }
                 printf("\n");
         }
         printf("Total Page Faults: %d\n",faults);
}

int search(int pno)
{
        int i;

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

```
                {
                                if(frames[i]==pno)
                                 return i;
                }

                return -1;
}

int get_mfu(int sp)
{
                int i,max_i,max=-9999;
                i=sp;
                do
                {
                                 if(count[i]>max)
                                 {
                                              max = count[i];
                                                max_i = i;
                                 }
                                 i=(i+1)%n;
                }while(i!=sp);
                return max_i;
}

void mfu()
{
                int i,j,k;

                for(i=0;i<m && sp<n;i++)
                {
                                 k=search(ref[i]);
                                 if(k==-1)
                 {
                                  frames[sp]=ref[i];
                                count[sp]++;
                                faults++;
                                  sp++;
                                  for(j=0;j<n;j++)
                                   mem[j][i]=frames[j];
                }
                 else
                 count[k]++;

 }
                sp=0;
                for(;i<m;i++)
                {
                                k = search(ref[i]);
                                if(k==-1)
                                {
                                                sp = get_mfu(sp);
                                                frames[sp] = ref[i];
                                               count[sp]=1;
                                               faults++;
                                               sp = (sp+1)%n;
                                              for(j=0;j<n;j++)
                                                 mem[j][i] = frames[j];
                               }
                                else
                                 count[k]++;
                }
}


int main()
{
                accept();
                mfu();
                disp();
                return 0;
}
```

Output:

```
Enter no.of frames:3

Enter no.of references:12

Enter reference string:

[0]=1

[1]=2

[2]=1

[3]=3

[4]=4

[5]=2
```

```
[6]=3

[7]=1

[8]=2

[9]=3

[10]=4

[11]=5

1  2  1  3  4  2  3  1  2  3  4  5


 1  1     1  4      4  4  3  3  3

    2    2  2      1  1  1  4  4

        3  3      3  2  2  2  5

Total Page Faults: 9
```