OS ASSIGNMENT 3

1) FIFO

CODE :

```
#include<stdio.h>

int main(){
   int pages, frames, pfault = 0;

   printf("Enter the number of pages : \n");
   scanf("%d",&pages);

   // int refString[] = {7,3,5,8,5,8,3,6,7,3,6,7,8,5,3};
   // int refString[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1};
   // int refString[] = {1,2,3,4,1,2,5,1,2,3,4,5};
   // pages = 20;

   int refString[pages];
   printf("Enter the Reference String : \n");
   for (int i = 0; i < pages; ++i)
      scanf("%d",&refString[i]);

   printf("Enter the number of frames : \n");
   scanf("%d",&frames);

   int frarr[frames];
   for (int i = 0; i < frames; ++i)
      frarr[i] = -9; //any neg number works


   printf("Pages\t");
   for (int i = 0; i < frames; ++i)
         printf("Frame%d\t",i);
```

```
    //FIFO
    for (int i = 0; i < pages; ++i)
    {
        int fl = 0; //flag
            for (int j = 0; j < frames; ++j)
                if(refString[i] ==  frarr[j]){
                    fl = 1;
                    break;
                }

        if (!fl){
            pfault++;
            int ind = (pfault - 1) %frames;
            frarr[ind] = refString[i];
        }

        printf("\n%d\t",refString[i]);
        for (int j = 0; j < frames; ++j)
        {
            if(frarr[j]==-9) printf(" -- \t");
            else printf("  %d \t",frarr[j]);
        }
        if(!fl) printf("\tPage Fault");
    }

    printf("\n\nThe total number of page faults : %d.\n",pfault);
    return 0;
}




/*
OUTPUT :
```

```
------------

Enter the number of frames :
3
Pages   Frame0  Frame1  Frame2
7        7       --      --          Page Fault
0        7       0       --          Page Fault
1        7       0       1           Page Fault
2        2       0       1           Page Fault
0        2       0       1
3        2       3       1           Page Fault
0        2       3       0           Page Fault
4        4       3       0           Page Fault
2        4       2       0           Page Fault
3        4       2       3           Page Fault
0        0       2       3           Page Fault
3        0       2       3
2        0       2       3
1        0       1       3           Page Fault
2        0       1       2           Page Fault
0        0       1       2
1        0       1       2
7        7       1       2           Page Fault
0        7       0       2           Page Fault
1        7       0       1           Page Fault

The total number of page faults : 15.
*/
```

2) OPTIMAL

CODE :

```
#include<stdio.h>

int optimal(int refString[], int frarr[], int pages, int frames, int
i,int pfault){
	if(pfault <= frames){
		return pfault-1;
	}

	int max = 0;
	int ind = -1;
	int k;

	//getting farthest page from the right
	for(int j = 0; j < frames; j++){
		int dist = 0;
		for(k = i; k < pages; k++){
			dist++;
			// printf("frarr[%d] (%d) == refString[%d]
(%d)\n",j,frarr[j],k,refString[k]);
			if(frarr[j] == refString[k]){

				if(dist > max){
					max = dist;
					ind = j;
				}
				break;
			}
		}

		// printf("ind : %d\tk : %d\n",ind,k);
		if(k == pages) return j; //page never referenced again
	}
```

```
         //i.e if no page in frames is referenced in string then
        //return any frame here we return frame 0,
        //else return the frame farthest page in reference string.
        if(ind == -1) return 0;
        else return ind;
}

int main(){
        int pages, frames, pfault = 0;

        printf("Enter the number of pages : \n");
        scanf("%d",&pages);

        int refString[pages];
        printf("Enter the Reference String : \n");
        for (int i = 0; i < pages; ++i)
                scanf("%d",&refString[i]);

        // int refString[] = {7,3,5,8,5,8,3,6,7,3,6,7,8,5,3};
    // int refString[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1};

    // pages = 20;

        printf("Enter the number of frames : \n");
        scanf("%d",&frames);

        int frarr[frames];
        for (int i = 0; i < frames; ++i)
        frarr[i] = -9; //any neg number works



    printf("Pages\t");
    for (int i = 0; i < frames; ++i)
        printf("Frame%d\t",i);
```

```
   //OPTIMAL
   for(int i = 0; i < pages; i++){
      int fl = 0;

      for(int j = 0; j < frames; j++){
            if(refString[i] ==  frarr[j]){
                  fl = 1;
                  break;
            }
      }

      if(!fl){
            pfault++;
            int ind = 
optimal(refString,frarr,pages,frames,i+1,pfault);
            frarr[ind] = refString[i] ;
      }

      printf("\n%d\t",refString[i]);
      for (int j = 0; j < frames; ++j)
      {
            if(frarr[j]==-9) printf(" -- \t");
            else printf("  %d \t",frarr[j]);
      }
      if(!fl) printf("\tPage Fault");
   }

   printf("\n\nThe total number of page faults : %d.\n",pfault);
   return 0;
}


/*
OUTPUT :
------------
```

```
Enter the number of frames :
3
Pages   Frame0  Frame1  Frame2
7        7      --      --          Page Fault
0        7       0      --          Page Fault
1        7       0       1          Page Fault
2        2       0       1          Page Fault
0        2       0       1
3        2       0       3          Page Fault
0        2       0       3
4        2       4       3          Page Fault
2        2       4       3
3        2       4       3
0        2       0       3          Page Fault
3        2       0       3
2        2       0       3
1        2       0       1          Page Fault
2        2       0       1
0        2       0       1
1        2       0       1
7        7       0       1          Page Fault
0        7       0       1
1        7       0       1

The total number of page faults : 9.
*/
```

3) LRU

```
#include<stdio.h>
```

```
int lru(int refString[], int frarr[], int pages, int frames, int i,int
pfault){
        if(pfault <= frames){
                return pfault-1;
        }

        int max = 0;
        int ind = -1;
        int k;

        //getting farthest page from the left
        for(int j = 0; j < frames; j++){
                int dist = 0;
                for(k = i; k >= 0 ; k--){
                        dist++;
                        // printf("frarr[%d] (%d) == refString[%d]
(%d)\n",j,frarr[j],k,refString[k]);
                        if(frarr[j] == refString[k]){
                                if(dist > max){
                                        max = dist;
                                        ind = j;
                                }
                                break;
                        }
                }
                // printf("ind : %d\tk : %d\n",ind,k);
        }
        return ind;
}

int main(){
        int pages, frames, pfault = 0;

        printf("Enter the number of pages : \n");
```

```
    scanf("%d",&pages);

    int refString[pages];
    printf("Enter the Reference String : \n");
    for (int i = 0; i < pages; ++i)
        scanf("%d",&refString[i]);

    // int refString[] = {7,3,5,8,5,8,3,6,7,3,6,7,8,5,3};
// int refString[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1};
// pages = 20;

    printf("Enter the number of frames : \n");
    scanf("%d",&frames);

    int frarr[frames];
    for (int i = 0; i < frames; ++i)
    frarr[i] = -9; //any neg number works


printf("Pages\t");
for (int i = 0; i < frames; ++i)
    printf("Frame%d\t",i);

//LRU
for(int i = 0; i < pages; i++){
    int fl = 0;

    for(int j = 0; j < frames; j++){
        if(refString[i] ==  frarr[j]){
            fl = 1;
            break;
        }
    }

    if(!fl){
```

```
                pfault++;
                int ind = lru(refString,frarr,pages,frames,i-1,pfault);
                frarr[ind] = refString[i] ;
        }

        printf("\n%d\t",refString[i]);
        for (int j = 0; j < frames; ++j)
        {
                if(frarr[j]==-9) printf(" -- \t");
                else printf("  %d \t",frarr[j]);
        }
        if(!fl) printf("\tPage Fault");
    }

    printf("\n\nThe total number of page faults : %d.\n",pfault);
    return 0;
}
```

```
/*
OUTPUT
------
Enter the number of frames :
```

```
3
Pages   Frame0  Frame1  Frame2
7        7      --      --          Page Fault
0        7       0      --          Page Fault
1        7       0       1          Page Fault
2        2       0       1          Page Fault
0        2       0       1
3        2       0       3          Page Fault
0        2       0       3
4        4       0       3          Page Fault
2        4       0       2          Page Fault
3        4       3       2          Page Fault
0        0       3       2          Page Fault
3        0       3       2
2        0       3       2
1        1       3       2          Page Fault
2        1       3       2
0        1       0       2          Page Fault
1        1       0       2
7        1       0       7          Page Fault
0        1       0       7
1        1       0       7

The total number of page faults : 12.

*/
```