1.optimal page rep alg

```
#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;
}
```

2. LRU page Replace

```
#include<bits/stdc++.h>

using namespace std;




int findlru(int time[],int n){

int minimum=time[0],pos=0;

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


    if(minimum>time[i]){

        minimum=time[i];

        pos=i;

    }

}


return pos;

}

 int main(){


 int pages[100],frame[10],time[10],num_frame,num_pages,len,counter=0,falut=0,pos;

 cout<<" Enter the number of frame"<<endl;

 cin>>num_frame;


 cout<<" enter the pages  length : "<<endl;

 cin>>len;

 cout<<" Enter the number of pages "<<endl;


 for(int i=0;i<len;i++)

 {

     cin>>pages[i];
```

```
}



for(int i=0;i<num_frame;i++){
  frame[i]= -1;
}

int flag1,flag2;

for(int i=0;i<len;i++){
  flag1=flag2=0;
  for(int j=0;j<num_frame;j++)
  {
    if(frame[j]== pages[i])
    {
      counter++;
      time[j]=counter;
      flag1=1;
      flag2=1;
     break;
    }

  }
  if(flag1==0){

     for(int j=0;j<num_frame;j++)
     {
        if(frame[j]==-1)
        {
           counter++;
           falut++;
```

```
                frame[j]=pages[i];

                time[j]=counter;

                flag2=1;

                break;

            }

        }


    }

    if(flag2==0){


        pos=findlru(time,num_frame);


        frame[pos]=pages[i];

        counter++;

        falut++;

        time[pos]=counter;

    }

cout<<"\n";

    for(int j=0;j<num_frame;j++){

        cout<<frame[j]<<"\t";

    }


     }


 cout<<" total page fault is "<<falut<<endl;

 return 0;

 }
```

3.Fifo page replace

```
#include <bits/stdc++.h>

using namespace std;


const int N=100005;


int n;

int frame_size;

int pages[N];

int mark[N];


void fifo_page_replacement(void)

{

    queue<int> Q;

    int page_faults=0;


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

    {

        if(mark[pages[i]]==true)

        {

            cout<<"Reference to page "<<pages[i]<<" did not cause a page fault\n";

        }

        else

        {

            Q.push(pages[i]);

            mark[pages[i]]=true;

            if(Q.size()>frame_size)

            {

                int p= Q.front();

                mark[p]=false;

                Q.pop();

            }
            page_faults++;

            cout<<"Reference to page "<<pages[i]<<" caused a page fault\n";

        }


    }

    cout<<"\nTotal Page Faults: "<<page_faults<<endl;

        cout<<"\nTotal Page hit: "<<n-page_faults<<endl;




    return;

}


int main()

{

    cout<<"Number of Frames: ";

    cin>>frame_size;


    cout<<"Page Reference Stream Length: ";

    cin>>n;


    cout<<"Page Reference Stream:\n";

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

        cin>>pages[i];


    fifo_page_replacement();


    return 0;

}\
```

4. round_robin page replace algoritm

```
while(true){
```

```
for(i=0,count=0;i<n;i++){

   int temp=q_t;

   if(rem_bst[i]==0){
         count++;
   continue;

   }

   if(rem_bst[i]>q_t){

      rem_bst[i]=rem_bst[i]-q_t;

   }
   else

   if(rem_bst[i]>=0){
      temp=rem_bst[i];
      rem_bst[i]=0;
   }
   ex=ex+temp;
   tat[i]=ex-art[i];


}

if(n==count)
   break;
```

```
}
```