OPERATING SYSTEM

LAB TASK 4

By: Sparsh Arya
Registration Number:17BEC0656
Slot: L7+L8

Page Replacement Algorithms

---

1. Code for FIFO Page Replacement Algorithm.

```
#include<stdio.h>
int main()
{
        int i,j,n,a[50],frame[10],no,k,avail,count=0;
      printf("\n ENTER THE NUMBER OF PAGES:\n");
        scanf("%d",&n);
      printf("\n ENTER THE PAGE NUMBER :\n");
      for(i=1;i<=n;i++)
      scanf("%d",&a[i]);
      printf("\n ENTER THE NUMBER OF FRAMES :");
      scanf("%d",&no);
        for(i=0;i<no;i++)
      frame[i]= -1;
              j=0;
              printf("\tref string\t page frames\n");
for(i=1;i<=n;i++)
              {
                      printf("%d\t\t",a[i]);
                      avail=0;
                      for(k=0;k<no;k++)
if(frame[k]==a[i])
                              avail=1;
                      if (avail==0)
                      {
                              frame[j]=a[i];
                              j=(j+1)%no;
                              count++;
                              for(k=0;k<no;k++)
                              printf("%d\t",frame[k]);
        }
                      printf("\n");
        }
              printf("Page Fault Is %d",count);
              return 0;
}
```

---

2. Code for LRU Page Replacement Algorithm

```
#include<stdio.h>
 int findLRU(int time[], int n){
   int i, minimum = time[0], pos = 0;
 for(i = 1; i < n; ++i){
       if(time[i] < minimum){
          minimum = time[i];
          pos = i;
       }
   }
   return pos;
}

int main()
{
   int no_of_frames, no_of_pages, frames[10], pages[30], counter = 0, time[10], flag1, flag2, i, j, pos, faults = 0;
   printf("Enter number of frames: ");
   scanf("%d", &no_of_frames);

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

   printf("Enter 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]){
         counter++;
         time[j] = counter;
            flag1 = flag2 = 1;
            break;
        }
   }

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

```
            }
        }

        if(flag2 == 0){
            pos = findLRU(time, no_of_frames);
            counter++;
            faults++;
            frames[pos] = pages[i];
            time[pos] = counter;
        }

        printf("\n");

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

    printf("\n\nTotal Page Faults = %d", faults);
  return 0;
}
```

3. Code for LFU Page Replacement Algorithm

```
#include<stdio.h>

int main()
{
    int total_frames, total_pages, hit = 0;
    int pages[25], frame[10], arr[25], time[25];
    int m, n, page, flag, k, minimum_time, temp;
    printf("Enter Total Number of Pages:\t");
    scanf("%d", &total_pages);
    printf("Enter Total Number of Frames:\t");
    scanf("%d", &total_frames);
    for(m = 0; m < total_frames; m++)
    {
        frame[m] = -1;
    }
    for(m = 0; m < 25; m++)
    {
        arr[m] = 0;
    }
    printf("Enter Values of Reference String\n");
    for(m = 0; m < total_pages; m++)
    {
        printf("Enter Value No.[%d]:\t", m + 1);
        scanf("%d", &pages[m]);
    }
    printf("\n");
    for(m = 0; m < total_pages; m++)
    {
        arr[pages[m]]++;
        time[pages[m]] = m;
        flag = 1;
        k = frame[0];
        for(n = 0; n < total_frames; n++)
        {
            if(frame[n] == -1 || frame[n] == pages[m])
            {
                if(frame[n] != -1)
                {
                    hit++;
                }
                flag = 0;
                frame[n] = pages[m];
                break;
            }
            if(arr[k] > arr[frame[n]])
            {
                k = frame[n];
            }
```

```
        }
        if(flag)
        {
            minimum_time = 25;
            for(n = 0; n < total_frames; n++)
            {
                if(arr[frame[n]] == arr[k] && time[frame[n]] < minimum_time)
                {
                    temp = n;
                    minimum_time = time[frame[n]];
                }
            }
            arr[frame[temp]] = 0;
            frame[temp] = pages[m];
        }
        for(n = 0; n < total_frames; n++)
        {
            printf("%d\t", frame[n]);
        }
        printf("\n");
    }
    printf("Page Hit:\t%d\n", hit);
    return 0;
}
```

4. Code for Optimal Page Replacement Algorithm.

```
#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;
}
```

File Allocation Strategies

1. Sequential File Allocation Strategy.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
   int st[20],b[20],b1[20],ch,i,j,n,blocks[20][20],sz[20];
   char F[20][20],S[20];
   clrscr();
   printf("\n Enter no. of Files ::");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      printf("\n Enter file %d name ::",i+1);

      scanf("%s",&F[i]);
      printf("\n Enter file%d size(in kb)::",i+1);
      scanf("%d",&sz[i]);
      printf("\n Enter Starting block of %d::",i+1);
      scanf("%d",&st[i]);
      printf("\n Enter blocksize of File%d(in bytes)::",i+1);
      scanf("%d",&b[i]);
   }
   for(i=0;i<n;i++)
      b1[i]=(sz[i]*1024)/b[i];
   for(i=0;i<n;i++)
   {
      for(j=0;j<b1[i];j++)
         blocks[i][j]=st[i]+j;
   }
   do
   {
      printf("\nEnter the Filename ::");
      scanf("%s",S);
      for(i=0;i<n;i++)
      {
         if(strcmp(S,F[i])==0)
         {
            printf("\nFname\tStart\tNblocks\tBlocks\n");
            printf("\n----------------------------------------------\n");
            printf("\n%s\t%d\t%d\t",F[i],st[i],b1[i]);
            for(j=0;j<b1[i];j++)
               printf("%d->",blocks[i][j]);
         }
```

```
        }
        printf("\n-----------------------------------------------\n");
        printf("\nDo U want to continue ::(Y:n)");
        scanf("%d",&ch);
        if(ch!=1)
            break;
    }while(1);
}
```

2. Code for Indexed File Allocation Strategy.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int n;
void main()
{
   int b[20],b1[20],i,j,blocks[20][20],sz[20];
   char F[20][20],S[20],ch;
   clrscr();
   printf("\n Enter no. of Files ::");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      printf("\n Enter file %d name ::",i+1);
      scanf("%s",&F[i]);
      printf("\n Enter file%d size(in kb)::",i+1);
      scanf("%d",&sz[i]);
      printf("\n Enter blocksize of File%d(in bytes)::",i+1);
      scanf("%d",&b[i]);
   }
   for(i=0;i<n;i++)
   {
      b1[i]=(sz[i]*1024)/b[i];
      printf("\n\nEnter blocks for file%d",i+1);
      for(j=0;j<b1[i];j++)
      {
         printf("\n Enter the %dblock ::",j+1);
         scanf("%d",&blocks[i][j]);
      }
   }
   do
   {
      printf("\nEnter the Filename ::");
      scanf("%s",&S);
      for(i=0;i<n;i++)
      {
         if(strcmp(F[i],S)==0)
         {
            printf("\nFname\tFsize\tBsize\tNblocks\tBlocks\n");
            printf("\n-----------------------------------------------\n");
            printf("\n%s\t%d\t%d\t%d\t",F[i],sz[i],b[i],b1[i]);
            for(j=0;j<b1[i];j++)
               printf("%d->",blocks[i][j]);
         }
      }
      printf("\n-----------------------------------------------\n");
      printf("\nDo U want to continue ::(Y:n)");
      scanf("%d",&ch);
   }while(ch!=0);
}
```

3. Code for Linked File Allocation Strategy.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int n;
void main()
{
   int b[20],b1[20],i,j,blocks[20][20],sz[20];
   char F[20][20],S[20],ch;
   int sb[20],eb[20],x;
   clrscr();
   printf("\n Enter no. of Files ::");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      printf("\n Enter file %d name ::",i+1);
      scanf("%s",&F[i]);
      printf("\n Enter file%d size(in kb)::",i+1);
      scanf("%d",&sz[i]);
      printf("\n Enter blocksize of File%d(in bytes)::",i+1);
      scanf("%d",&b[i]);
   }
   for(i=0;i<n;i++)
   {
      b1[i]=(sz[i]*1024)/b[i];
      printf("\n Enter Starting block of file%d::",i+1);
      scanf("%d",&sb[i]);
      printf("\n Enter Ending block of file%d::",i+1);
      scanf("%d",&eb[i]);
      printf("\nEnter blocks for file%d::\n",i+1);
      for(j=0;j<b1[i]-2;)
      {
         printf("\n Enter the %dblock ::",j+1);
         scanf("%d",&x);
         if(x>sb[i]&&x<eb[i])
         {
            blocks[i][j]=x;
            j++;
         }
         else
            printf("\n Invalid block::");
      }
   }
   do
   {
      printf("\nEnter the Filename ::");
      scanf("%s",&S);
      for(i=0;i<n;i++)
      {
         if(strcmp(F[i],S)==0)
         {
```

```
            printf("\nFname\tFsize\tBsize\tNblocks\tBlocks\n");
            printf("\n-----------------------------------------------\n");
            printf("\n%s\t%d\t%d\t%d\t",F[i],sz[i],b[i],b1[i]);
            printf("%d->",sb[i]);
            for(j=0;j<b1[i]-2;j++)
               printf("%d->",blocks[i][j]);
            printf("%d->",eb[i]);
          }
       }
       printf("\n-----------------------------------------------\n");
       printf("\nDo U want to continue (Y:n)::");
       scanf("%d",&ch);
    }while(ch!=0);
}
```

x----------x