OS Lab Assignment Prashanth.S(19MID0020)

Memory Replacement Algorithm

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

int main()
{
	int ch;
	printf("1: First Fit\n");
	printf("2: Best Fit\n");
	printf("3: Worst Fit\n");
	do
	{
		printf("Enter the choice: ");
		scanf("%d",&ch);
		switch(ch)
		{
		case 1:
			first_fit();
			break;
		case 2:
			best_fit();
			break;
		case 3:
			worst_fit();
			break;
		default:
			exit(0);

		}ch++;
	}while(ch<7);
	return 0;
}

void first_fit( )
{
	int nb,np,i,j;
	printf("Enter the no.of.blocks:");
	scanf("%d",&nb);
	printf("Enter the no.of.process:");
	scanf("%d",&np);
	int b[nb],p[np],flag[np];
	printf("Enter the block size one by one :");
	for(  i=0;i<nb;i++)
	{
		scanf("%d",&b[i]);
	}
	printf("Enter the process size one by one :");
	for(  i=0;i<np;i++)
	{
		scanf("%d",&p[i]);
	}
	for(i=0;i<np;i++)
	{
		flag[i]=0;
	}
```

```
55      for(i=0;i<np;i++)
56      {
57            for(j=0;j<nb;j++)
58            {
59                  if  (flag[i]==0 && b[j]>=p[i])
60                  {
61
62                        flag[i]=j+1;
63                        b[j]=b[j]-p[i];
64                        break;
65                  }
66            }
67      }
68
69      for (i=0;i<np;i++)
70      {
71            if (flag[i]>0)
72            {
73                  printf("The process [%d] is allocated in block [%d] \n",i+1,flag[i]);
74            }
75            else
76            {
77                  printf("the process [%d] is not allocated in any blocks\n",i+1);
78            }
79      }
80 }
```

```
81 void best_fit( )
82 {
83       int fragment[20],b[20],p[20],i,j,nb,np,tem,low=9999;
84         static int barray[20],parray[20];
85         printf("Enter the number of processes:");
86       scanf("%d",&np);
87         printf("\nEnter the number of blocks:");
88         scanf("%d",&nb);
89         printf("\nEnter the size of the blocks:-\n");
90         for(i=1;i<=nb;i++)
91         {
92
93                 printf("Block no.%d:",i);
94
95                         scanf("%d",&b[i]);
96         }
97         printf("\nEnter the size of the processes :-\n");
98         for(i=1;i<=np;i++)
99      {
100
101         printf("Process no.%d:",i);
102
103         scanf("%d",&p[i]);
104     }
```

```
105         for(i=1;i<=np;i++)
106         {
107                 for(j=1;j<=nb;j++)
108                 {
109
110                         if(barray[j]!=1)
111                         {
112                                 tem=b[j]-p[i];
113                                 if(tem>=0)
114                                         if(low>tem)
115                                         {
116                                                 parray[i]=j;
117                                                 low=tem;
118                                         }
119                         }
120                 }
121
122
123                 fragment[i]=low;
124                 barray[parray[i]]=1;
125                 low=10000;
126         }
127
128
129 printf("\nProcess_number \tProcess_size\tBlock_number \tBlock_size\tFragment");
130         for(i=1;i<=np && parray[i]!=0;i++)
131
132                 printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d",i,p[i],parray[i],b[parray[i]],fragment[i]);
133 }
134
135 void worst_fit()
136 {
137        int bsize[20];
138     int psize[20];
139     int i,j,k,l,bn,pn;
140     int allocate[20];
141     int frag[20];
142     int max[20];
143     printf("Enter the total number of blocks:");
144     scanf("%d",&bn);
145     printf("Enter the sizes of blocks:");
146     for(i=0;i<bn;i++)
147     {
148         scanf("%d",&bsize[i]);
149     }
150     printf("Enter the total number of process:");
151      scanf("%d",&pn);
152     printf("Enter the sizes of process:");
```

```
    printf("Enter the sizes of process:");
    for(i=0;i<pn;i++)
    {
        scanf("%d",&psize[i]);
    }
    for(i=0;i<bn;i++)
    {
        allocate[i]=0;
    }
    for(i=0;i<pn;i++)
    {
        for(j=0;j<bn;j++)
        {
            frag[j]=bsize[j]-psize[i];
        }
        max[i]=frag[0];
        for(k=0;k<bn;k++)
            if (frag[k] > max[i])
            max[i] = frag[k];
        for(l=0;l<bn;l++)
        {
            if(frag[l]==max[i]&&allocate[i]==0&&bsize[l]>psize[i])
            {
                allocate[i]=l;
                bsize[l]=bsize[l]-psize[i];

        }

      }
   }
    for (i=0;i<pn;i++)
     {
          if (allocate[i]>0)
          {
               printf("The process [%d] is allocated in block [%d] \n",i,allocate[i]);
          }
          else
          {
               printf("the process [%d] is not allocated in any blocks\n",i);
          }
     }
}
```

```
1: First Fit
2: Best Fit
3: Worst Fit

Enter the choice: 2
Enter the number of processes:4

Enter the number of blocks:5

Enter the size of the blocks:-
Block no.1:100
Block no.2:500
Block no.3:200
Block no.4:300
Block no.5:600

Enter the size of the processes :-
Process no.1:212
Process no.2:417
Process no.3:112
Process no.4:426

Process_number  Process_size    Block_number    Block_size      Fragment
1               212             4               300             88
2               417             2               500             83
3               112             3               200             88
4               426             5               600             174
Enter the choice: 1
Enter the no.of.blocks:5
Enter the no.of.process:4
Enter the block size one by one :100 500 200 300 600
Enter the process size one by one :212 417 112 426
The process [1] is allocated in block [2]
The process [2] is allocated in block [5]
The process [3] is allocated in block [2]
the process [4] is not allocated in any blocks

Enter the choice: 0

--------------------------------
Process exited after 94.41 seconds with return value 0
Press any key to continue . . .
```

1)FIFO (First in First out)

Open ▾ fifo.c ~/Page replacement algorithm

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

int main()
{
	int reference_string[10], page_faults = 0, m, n, s, pages, frames;
	printf("\nEnter Total Number of Pages : ");
	scanf("%d", &pages);

	printf("\nEnter values of Reference String:\n");
	for(m=0;m<pages;m++)
	{
		printf("Value No. [%d]:\t", m + 1);
		scanf("%d", &reference_string[m]);
	}

	printf("\nEnter Total Number of Frames : ");
	{ scanf("%d", &frames); }
	int temp[frames];
	for(m=0; m<frames;m++) { temp[m] = -1;  }
	for(m = 0; m < pages; m++)
	{
		s=0;
		for(n = 0;n<frames;n++)
		{
			if(reference_string[m] == temp[n])
			{
				s++;
				page_faults--;
			}
		}
		page_faults++;
		if((page_faults <= frames) && (s == 0)) { temp[m] = reference_string[m]; }
		else if(s==0)  { temp[(page_faults-1)%frames] = reference_string[m]; }

		printf("\n");
		for(n = 0; n < frames; n++) {  printf("%d\t", temp[n]);  }
	}
	printf("\nTotal Page Faults : %d\n", page_faults);
	return 0;
}
```

```
prashanth@prashanth-VirtualBox:~$ ./a.out

Enter Total Number of Pages : 10

Enter values of Reference String:
Value No. [1]:  1
Value No. [2]:  2
Value No. [3]:  3
Value No. [4]:  4
Value No. [5]:  2
Value No. [6]:  1
Value No. [7]:  5
Value No. [8]:  6
Value No. [9]:  2
Value No. [10]: 1

Enter Total Number of Frames : 3

1       -1      -1
1       2       -1
1       2       3
4       2       3
4       2       3
4       1       3
4       1       5
6       1       5
6       2       5
6       2       1
Total Page Faults : 9
```

2) Optimal Page replacement Algorithm

Open optimal.c

```c
#include<stdio.h>
int main()
{
    int no_of_frames, no_of_pages, frames[10], pages[30], temp[10];
    int 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:

```
prashanth@prashanth-VirtualBox:~$ gcc optimal.c
prashanth@prashanth-VirtualBox:~$ ./a.out
Enter number of frames: 3
Enter number of pages: 10
Enter page reference string: 2 3 4 6 7 2 7 3 2 5

2       -1      -1
2       3       -1
2       3       4
2       3       6
2       3       7
2       3       7
2       3       7
2       3       7
2       3       7
5       3       7

Total Page Faults = 6prashanth@prashanth-VirtualBox:~$
```

3) Least Recently Used Algorithm (LRU)

```c
#include<stdio.h>
int main()
{
int q[20],p[50],c=0,c1,d,f,i,j,k=0,n,r,t,b[20],c2[20];

printf("Enter no of frames:");
scanf("%d",&f);
printf("Enter no of pages:");
scanf("%d",&n);
printf("Enter the reference string:");
for(i=0;i<n;i++) { scanf("%d",&p[i]); }

q[k]=p[k];
printf("\n\t%d\n",q[k]);
c++;
k++;
for(i=1;i<n;i++)
    {
                c1=0;
                for(j=0;j<f;j++)
                {
                        if(p[i]!=q[j])
                        c1++;
                }
               if(c1==f)
               {
                       c++;
                       if(k<f)
                       {
                               q[k]=p[i];
                               k++;
                               for(j=0;j<k;j++)
                               printf("\t%d",q[j]);
                               printf("\n");
                       }
               else
               {
                       for(r=0;r<f;r++)
                       {
                               c2[r]=0;
                               for(j=i-1;j<n;j--)
                               {
                                       if(q[r]!=p[j])
                                       c2[r]++;
                                       else
                                       break;
                               }
                       }
                       for(r=0;r<f;r++)
                       b[r]=c2[r];
```

```
                        for(r=0;r<f;r++)
                        {
                                for(j=r;j<f;j++)
                                        {
                                                if(b[r]<b[j])
                                                        {
                                                                t=b[r];
                                                                b[r]=b[j];
                                                                b[j]=t;
                             }
                                        }
                        }

                        for(r=0;r<f;r++)
                        {
                                if(c2[r]==b[0])
                                q[r]=p[i];
                                printf("\t%d",q[r]);
                        }
                                printf("\n");
                }
        }
}
printf("\nThe no of page faults is %d",c);
}
```

```
prashanth@prashanth-VirtualBox:~$ gcc lru.c
prashanth@prashanth-VirtualBox:~$ ./a.out
Enter no of frames:3
Enter no of pages:10
Enter the reference string:1 2 3 4 9 2 3 8 6 3

        1
        1       2
        1       2       3
        4       2       3
        4       9       3
        4       9       2
        3       9       2
        3       8       2
        3       8       6

The no of page faults is 9prashanth@prashanth-VirtualBox:~$
```