OPERATING SYSTEMS LAB - PRACTICAL 9

Name - Sakshi Soni
Roll No - 13

AIM -

Write C programs to implement different disk scheduling algorithms, page replacement algorithms and to demonstrate different memory management schemes.

PROGRAM AND OUTPUT -

Program 9A - Worst Fit Memory Management Scheme

```
#include <stdio.h>

#define MAX_BLOCKS 100
#define MAX_FILES 100

void worstFit(int blockSize[], int m, int fileSize[], int n)
{
    int allocation[MAX_FILES];

    for (int i = 0; i < n; i++)
    {
        int wstIdx = -1;
        for (int j = 0; j < m; j++)
        {
            if (blockSize[j] >= fileSize[i])
            {
                if (wstIdx == -1)
                {
```

```
                wstIdx = j;
            }
            else if (blockSize[j] > blockSize[wstIdx])
            {
                wstIdx = j;
            }
        }
    }

    if (wstIdx != -1)
    {
        allocation[i] = wstIdx;
        blockSize[wstIdx] -= fileSize[i];
    }
    else
    {
        allocation[i] = -1;
    }
}

printf("\nFile No.\tFile Size\tBlock No.\n");
for (int i = 0; i < n; i++)
{
    printf(" %d\t\t%d\t\t", i + 1, fileSize[i]);
    if (allocation[i] != -1)
    {
        printf("%d\n", allocation[i] + 1);
    }
    else
    {
        printf("Not Allocated\n");
    }
}
}
```

```
int main()
{
    int blockSize[MAX_BLOCKS], fileSize[MAX_FILES];
    int m, n;

    printf("Enter the number of memory blocks: ");
    scanf("%d", &m);

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

    printf("Enter the sizes of the memory blocks:\n");
    for (int i = 0; i < m; i++)
    {
        scanf("%d", &blockSize[i]);
    }

    printf("Enter the sizes of the files:\n");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &fileSize[i]);
    }

   worstFit(blockSize, m, fileSize, n);

    return 0;
}
```

```
winter@windows:~/OS/prac9$ gedit worst_fit.c
^C
winter@windows:~/OS/prac9$ gcc worst_fit.c
winter@windows:~/OS/prac9$ ./a.out
Enter the number of memory blocks: 5
Enter the number of files: 4
Enter the sizes of the memory blocks:
100
500
200
300
600
Enter the sizes of the files:
212
417
112
426

File No.        File Size       Block No.
 1              212             5
 2              417             2
 3              112             5
 4              426             Not Allocated
winter@windows:~/OS/prac9$
```

Program 9B - FIFO Page Replacement Algorithm

```
#include <stdio.h>
#define MAX_FRAMES 10
int main()
{
        int frames[MAX_FRAMES], faults = 0;
        int num_frames, num_pages;

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

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

```
int pages[num_pages];
printf("Enter the page reference string:\n");
for (int i = 0; i < num_pages; i++)
{
scanf("%d", &pages[i]);
}

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

int next_frame = 0;
for (int i = 0; i < num_pages; i++)
{
int page = pages[i];
int found = 0;

// Check if the page is already in a frame
for (int j = 0; j < num_frames; j++)
{
if (frames[j] == page)
{
        found = 1;
        break;
}
}

if (!found)
{
frames[next_frame] = page;
next_frame = (next_frame + 1) % num_frames;
faults++;
}
```

```
        printf("Page reference: %d - Frames: ", page);
        for (int j = 0; j < num_frames; j++)
        {
        printf("%d ", frames[j]);
        }
        printf("\n");
        }

        printf("Total page faults: %d\n", faults);

        return 0;
}
```

```
winter@windows:~/OS/prac9$ gedit FIFO_page.c
^C
winter@windows:~/OS/prac9$ gcc FIFO_page.c
winter@windows:~/OS/prac9$ ./a.out
Enter the number of frames: 3
Enter the number of pages: 7
Enter the page reference string:
1
2
3

4
5
1
2
Page reference: 1 - Frames: 1 -1 -1
Page reference: 2 - Frames: 1 2 -1
Page reference: 3 - Frames: 1 2 3
Page reference: 4 - Frames: 4 2 3
Page reference: 5 - Frames: 4 5 3
Page reference: 1 - Frames: 4 5 1
Page reference: 2 - Frames: 2 5 1
Total page faults: 7
winter@windows:~/OS/prac9$
```

Program 9C - SCAN Disk Scheduling Algorithm

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

// Function to sort an array in ascending order
void sort(int arr[], int n)
{
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

void scan(int tracks[], int n, int head, char direction)
{
    // Sort the tracks array in ascending order
    sort(tracks, n);

    int i, j, seek_count = 0;
    int distance, cur_track;

    int left[n], right[n];
    int left_count = 0, right_count = 0;

    // Divide the tracks into two arrays: left and right
    for (i = 0; i < n; i++)
    {
```

```
    if (tracks[i] < head)
        left[left_count++] = tracks[i];
    if (tracks[i] > head)
        right[right_count++] = tracks[i];
}

// Perform SCAN algorithm based on the direction
if (direction == 'l')
{
    for (i = left_count - 1; i >= 0; i--)
    {
        cur_track = left[i];
        printf("%d -> ", cur_track);
        seek_count += abs(cur_track - head);
        head = cur_track;
    }

    printf("0 -> ");

    for (j = 0; j < right_count; j++)
    {
        cur_track = right[j];
        printf("%d -> ", cur_track);
        seek_count += abs(cur_track - head);
        head = cur_track;
    }
}
else if (direction == 'r')
{
    for (j = 0; j < right_count; j++)
    {
        cur_track = right[j];
        printf("%d -> ", cur_track);
        seek_count += abs(cur_track - head);
        head = cur_track;
```

```
        }

        printf("199 -> ");

        for (i = left_count - 1; i >= 0; i--)
        {
            cur_track = left[i];
            printf("%d -> ", cur_track);
            seek_count += abs(cur_track - head);
            head = cur_track;
        }
    }

    printf("\nTotal head movement: %d\n", seek_count);
}

int main()
{
    int n, head;
    char direction;

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

    int tracks[n];
    printf("Enter the track positions:\n");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &tracks[i]);
    }

    printf("Enter the head position: ");
    scanf("%d", &head);

    printf("Enter the direction (l for left, r for right): ");
```

```
    scanf(" %c", &direction);

    printf("\nHead movement trace:\n");
    scan(tracks, n, head, direction);

    return 0;
}
```

```
winter@windows:~/OS/prac9$ gedit FIFO_page.c
^C
winter@windows:~/OS/prac9$ gedit SCAN_disk.c
^C
winter@windows:~/OS/prac9$ gcc SCAN_disk.c
winter@windows:~/OS/prac9$ ./a.out
Enter the number of tracks: 7
Enter the track positions:
98
183
37
122
14
124
65
Enter the head position: 53
Enter the direction (l for left, r for right): r

Head movement trace:
65 -> 98 -> 122 -> 124 -> 183 -> 199 -> 37 -> 14 ->
Total head movement: 299
winter@windows:~/OS/prac9$
```

RESULT -

C programs to implement disk scheduling algorithms, page replacement algorithms and to demonstrate different memory management schemes have been implemented.