2CS5

Group2

Rishi Malik: 102017096

Naman Khurana: 102017099

Amol Sharma: 102017106

**UCS303 – Operating Systems**

**Lab 8**

Write a C / C++ program to simulate **disk scheduling algorithms**

a) FCFS

b) SSTF

c) SCAN

d) C-SCAN

e) LOOK

f) C-LOOK

1. **FCFS**

#include "stdio.h"

#include "stdlib.h"

#include "stdbool.h"

int main()

{

    int i,no\_of\_requests,initial\_head;

    printf("Enter the number of requests: ");

    scanf("%d",&no\_of\_requests);

    int request[no\_of\_requests];

    printf("Enter the requests: ");

    for (i = 0; i < no\_of\_requests; ++i)

    {

        scanf("%d",&request[i]);

    }

    printf("Enter initial position of head: ");

    scanf("%d",&initial\_head);

    int seek\_time=0;

    printf("%d -> ",initial\_head );

    for(i=0;i<no\_of\_requests;i++)

    {

        if(i == no\_of\_requests-1)

            printf("%d\n", request[i] );

        else

            printf("%d -> ", request[i] );

        seek\_time += abs(request[i] - initial\_head);

        initial\_head = request[i];

    }

    printf("Seek Time: %d\n", seek\_time);

}

**2)SSTF**

#include "stdio.h"

#include "stdlib.h"

#include "stdbool.h"

struct request

{

    int request\_track\_number;

    bool visited;

};

int main()

{

    int i,no\_of\_requests,initial\_head,limit,j,choice,previous\_head;

    printf("Enter the number of requests: ");

    scanf("%d",&no\_of\_requests);

    struct request req[no\_of\_requests];

    printf("Enter the requests: ");

    for (i = 0; i < no\_of\_requests; ++i)

    {

        scanf("%d",&req[i].request\_track\_number);

        req[i].visited = false;

    }

    printf("Enter initial position of R/W head: ");

    scanf("%d",&initial\_head);

    int seek\_time=0;

    printf("%d -> ",initial\_head );

    int n = no\_of\_requests;

    while(n)

    {

        int min = 1e9;

        int min\_track\_number, position;

        for(i=0;i<no\_of\_requests;i++)

        {

            if(abs(initial\_head - req[i].request\_track\_number) < min && req[i].visited == false)

            {

                min = abs(initial\_head - req[i].request\_track\_number);

                min\_track\_number = req[i].request\_track\_number;

                position = i;

            }

        }

        initial\_head = req[position].request\_track\_number;

        req[position].visited = true;

        printf("%d ->",min\_track\_number);

        seek\_time += min;

        n--;

    }

    printf("\nSeek Time: %d\n", seek\_time);

}

1. **SCAN**

|  |
| --- |
| #include "stdio.h" |
|  | #include "stdlib.h" |
|  | #include "stdbool.h" |
|  |  |
|  | struct request |
|  | { |
|  | int request\_track\_number; |
|  | bool visited; |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | int i,no\_of\_requests,initial\_head,limit,j,choice,previous\_head; |
|  | printf("Enter the number of requests: "); |
|  | scanf("%d",&no\_of\_requests); |
|  | struct request req[no\_of\_requests]; |
|  | printf("Enter the requests: "); |
|  | for (i = 0; i < no\_of\_requests; ++i) |
|  | { |
|  | scanf("%d",&req[i].request\_track\_number); |
|  | req[i].visited = false; |
|  | } |
|  | printf("Enter initial position of R/W head: "); |
|  | scanf("%d",&initial\_head); |
|  |  |
|  | printf("Enter the previous position of R/W head: "); |
|  | scanf("%d",&previous\_head); |
|  |  |
|  | printf("Enter the cylinder size: "); |
|  | scanf("%d",&limit); |
|  |  |
|  | if(previous\_head - initial\_head > 0 ) |
|  | { |
|  | choice = 2; |
|  | } |
|  | else |
|  | choice = 1; |
|  | //scanf("%d",&choice); |
|  | int seek\_time=0; |
|  | printf("%d -> ",initial\_head ); |
|  | if(choice == 1) |
|  | { |
|  | for(i=initial\_head;i<limit;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("%d -> ", limit-1); |
|  | seek\_time += abs(limit-1 - initial\_head); |
|  | initial\_head = limit-1; |
|  | for(i=initial\_head;i>=0;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | seek\_time += abs(initial\_head - 0); |
|  | printf("0 \n"); |
|  | } |
|  | else if(choice == 2) |
|  | { |
|  | for(i=initial\_head;i>=0;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("%d -> ", 0); |
|  | seek\_time += abs(0 - initial\_head); |
|  | initial\_head = 0; |
|  | for(i=initial\_head;i<limit;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | seek\_time += abs(limit-1 - initial\_head ); |
|  | printf("%d \n", limit-1); |
|  |  |
|  | } |
|  | printf("Seek Time: %d\n", seek\_time); |
|  | } |

1. **C-SCAN**

|  |
| --- |
| #include "stdio.h" |
|  | #include "stdlib.h" |
|  | #include "stdbool.h" |
|  |  |
|  | struct request |
|  | { |
|  | int request\_track\_number; |
|  | bool visited; |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | int i,no\_of\_requests,initial\_head,limit,j,choice,previous\_head; |
|  | printf("Enter the number of requests: "); |
|  | scanf("%d",&no\_of\_requests); |
|  | struct request req[no\_of\_requests]; |
|  | printf("Enter the requests: "); |
|  | for (i = 0; i < no\_of\_requests; ++i) |
|  | { |
|  | scanf("%d",&req[i].request\_track\_number); |
|  | req[i].visited = false; |
|  | } |
|  | printf("Enter initial position of R/W head: "); |
|  | scanf("%d",&initial\_head); |
|  |  |
|  | printf("Enter the previous position of R/W head: "); |
|  | scanf("%d",&previous\_head); |
|  |  |
|  | printf("Enter the cylinder size: "); |
|  | scanf("%d",&limit); |
|  |  |
|  | if(previous\_head - initial\_head > 0 ) |
|  | { |
|  | choice = 2; |
|  | } |
|  | else |
|  | choice = 1; |
|  | //scanf("%d",&choice); |
|  | int seek\_time=0; |
|  | printf("%d -> ",initial\_head ); |
|  | int cp\_initial\_head = initial\_head; |
|  | if(choice == 1) |
|  | { |
|  | for(i=initial\_head;i<limit;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("%d -> \n", limit-1); |
|  | seek\_time += abs(limit-1 - initial\_head); |
|  | initial\_head = 0; |
|  | for(i=0;i<cp\_initial\_head;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("\n"); |
|  | } |
|  | else if(choice == 2) |
|  | { |
|  | for(i=initial\_head;i>=0;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("%d -> ", 0 ); |
|  | seek\_time += abs(initial\_head - 0); |
|  | initial\_head = limit-1; |
|  | for(i=limit;i>cp\_initial\_head;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("\n"); |
|  | } |
|  | printf("Seek Time: %d\n", seek\_time); |
|  | } |

1. **LOOK**

|  |
| --- |
| #include "stdio.h" |
|  | #include "stdlib.h" |
|  | #include "stdbool.h" |
|  |  |
|  | struct request |
|  | { |
|  | int request\_track\_number; |
|  | bool visited; |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | int i,no\_of\_requests,initial\_head,limit,j,choice,previous\_head; |
|  | printf("Enter the number of requests: "); |
|  | scanf("%d",&no\_of\_requests); |
|  | struct request req[no\_of\_requests]; |
|  | printf("Enter the requests: "); |
|  | for (i = 0; i < no\_of\_requests; ++i) |
|  | { |
|  | scanf("%d",&req[i].request\_track\_number); |
|  | req[i].visited = false; |
|  | } |
|  | printf("Enter initial position of R/W head: "); |
|  | scanf("%d",&initial\_head); |
|  |  |
|  | printf("Enter the previous position of R/W head: "); |
|  | scanf("%d",&previous\_head); |
|  |  |
|  | printf("Enter the cylinder size: "); |
|  | scanf("%d",&limit); |
|  |  |
|  | if(previous\_head - initial\_head > 0 ) |
|  | { |
|  | choice = 2; |
|  | } |
|  | else |
|  | choice = 1; |
|  | //scanf("%d",&choice); |
|  | int seek\_time=0; |
|  | printf("%d -> ",initial\_head ); |
|  | if(choice == 1) |
|  | { |
|  | for(i=initial\_head;i<limit;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | for(i=initial\_head;i>=0;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("\n"); |
|  | } |
|  | else if(choice == 2) |
|  | { |
|  | for(i=initial\_head;i>=0;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | for(i=initial\_head;i<limit;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("\n"); |
|  | } |
|  | printf("Seek Time: %d\n", seek\_time); |
|  | } |

1. **C-LOOK**

|  |
| --- |
| #include "stdio.h" |
|  | #include "stdlib.h" |
|  | #include "stdbool.h" |
|  |  |
|  | struct request |
|  | { |
|  | int request\_track\_number; |
|  | bool visited; |
|  | }; |
|  |  |
|  | int main() |
|  | { |
|  | int i,no\_of\_requests,initial\_head,limit,j,choice,previous\_head; |
|  | printf("Enter the number of requests: "); |
|  | scanf("%d",&no\_of\_requests); |
|  | struct request req[no\_of\_requests]; |
|  | printf("Enter the requests: "); |
|  | for (i = 0; i < no\_of\_requests; ++i) |
|  | { |
|  | scanf("%d",&req[i].request\_track\_number); |
|  | req[i].visited = false; |
|  | } |
|  | printf("Enter initial position of R/W head: "); |
|  | scanf("%d",&initial\_head); |
|  |  |
|  | printf("Enter the previous position of R/W head: "); |
|  | scanf("%d",&previous\_head); |
|  |  |
|  | printf("Enter the cylinder size: "); |
|  | scanf("%d",&limit); |
|  |  |
|  | if(previous\_head - initial\_head > 0 ) |
|  | { |
|  | choice = 2; |
|  | } |
|  | else |
|  | choice = 1; |
|  | //scanf("%d",&choice); |
|  | int seek\_time=0; |
|  | printf("%d -> ",initial\_head ); |
|  | int cp\_initial\_head = initial\_head; |
|  | if(choice == 1) |
|  | { |
|  | for(i=initial\_head;i<limit;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | initial\_head = 0; |
|  | for(i=0;i<cp\_initial\_head;i++) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("\n"); |
|  | } |
|  | else if(choice == 2) |
|  | { |
|  | for(i=initial\_head;i>=0;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | initial\_head = limit-1; |
|  | for(i=limit;i>cp\_initial\_head;i--) |
|  | { |
|  | for(j=0;j<no\_of\_requests;j++) |
|  | { |
|  | if(req[j].request\_track\_number == i && req[j].visited == false) |
|  | { |
|  | printf("%d -> ", req[j].request\_track\_number); |
|  | req[j].visited = true; |
|  | seek\_time += abs(req[j].request\_track\_number - initial\_head); |
|  | initial\_head = req[j].request\_track\_number; |
|  | } |
|  | } |
|  | } |
|  | printf("\n"); |
|  | } |
|  | printf("Seek Time: %d\n", seek\_time); |
|  | } |