FCFS

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 200
void accept_requests(int requests[] , int nreqs)
{
   int i;
   printf("Enter requests :\n");
   for(i = 0 ; i< nreqs ; i++)
   {

        scanf("%d",&requests[i]);
        if(requests[i] >= MAX)
        {
        printf("INVALID");
           exit(0);
}
}
}
void fcfs(int head , int requests[] , int nreqs)
{
   int i, totalMovement = 0;

   printf("Sequence : ");
   for(i = 0 ; i<nreqs ; i++)
   {

      printf("%d - ",head);
      totalMovement += abs(head - requests[i]);
      head = requests[i];
   }
   printf("%d\n",head);
   printf("Total head movement : %d\n",totalMovement);
}

int main()
{
   int nreqs , head, i;
   printf("Enter the number of requests :");
   scanf("%d",&nreqs);

   int requests[nreqs];

   accept_requests(requests , nreqs);

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

   fcfs(head,requests,nreqs);
}
```

SSTF

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#define MAX 200
int i,j;
void sort(int arr[], int n)
{
    for (i = 0; i < n - 1; i++)
    {
        for (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 sstf(int head, int requests[], int nreqs)
{
    int total_movement = 0;
    int visited[100] = {0};
    int count = nreqs;
    int i = 0;

    sort(requests, nreqs);
    while (i < nreqs && requests[i] < head)
     {
        i++;
     }
    printf("Sequence: ");
 printf("%d - ",head);

    while (count > 0)
    {
        int left = i - 1;
        int right = i;

        while (left >= 0 && visited[left])
        {
            left--;
        }

        while (right < nreqs && visited[right])
         {
            right++;
        }

        int left_dist = (left >= 0) ? abs(requests[left] - head) : INT_MAX;
        int right_dist = (right < nreqs) ? abs(requests[right] - head) : INT_MAX;
```

```
        if (left_dist == INT_MAX && right_dist == INT_MAX) {
            break;
        } else if (left_dist < right_dist) {
            head = requests[left];
            visited[left] = 1;
            total_movement += left_dist;
            count--;
            i = left;
        } else {
            head = requests[right];
            visited[right] = 1;
            total_movement += right_dist;
            count--;
            i = right;
        }
        printf("%d -", head);
    }

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

int main() {
    int n,i;
    printf("Enter number of disk requests: ");
    scanf("%d", &n);

    int requests[100];
    printf("Enter requests: ");
    for (i = 0; i < n; i++)
    {
        scanf("%d", &requests[i]);
        if(requests[i] >= MAX)
        {
        printf("INVALID");
           exit(0);
    }
    }
    sort(requests, n);

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

    sstf(head, requests, n);

    return 0;
}
```

SCAN

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

```
#include<stdlib.h>

#define MAX 200

void sort(int arr[] , int n)
{
   int i, j, temp;
   for(i=0; i<n-1 ; i++)
   {
      for(j=0; j<n-i-1 ; j++)
      {
         if(arr[j] > arr[j+1])
         {
           temp = arr[j];
           arr[j] = arr[j+1];
           arr[j+1] = temp;
         }
      }
   }
}

void accept_requests(int requests[] , int nreqs)
{
   int i;
   printf("Enter requests : ");

  for (i = 0; i < nreqs; i++)
  {
        scanf("%d", &requests[i]);
        if(requests[i] >= MAX)
        {
            printf("INVALID");
            exit(0);
         }
    }

   sort(requests,nreqs);
}

void scan(int head , int direction , int requests[], int nreqs)
{
    int total_movement = 0;
    int count = nreqs;
    int visited[MAX] = {0};
    int i;

    if(direction == 1)
    {
        for(i=0; i<nreqs; i++)
        {
           if(requests[i]>=head)
              break;
        }
    }
    else
```

```
    {
       for(i=nreqs-1 ; i>=0 ; i--)
       {
          if(requests[i]<=head)
             break;
       }
    }

    printf("Sequence : %d",head);

    while(1)
    {
      if(i>= nreqs || i<0)
      {
          if(i<0)
          {
             total_movement += head;
             head =0;
             direction = 1;
             printf(" - %d",head);
          }
          else
          if(i>=nreqs)
            {
              total_movement += (MAX -1-head);
              head =MAX -1;
              direction = -1;
              printf(" - %d",head);
            }
          i += direction;
          continue;
      }

      if(visited[i] == 1)
      {
         i += direction;
         continue;
      }

      total_movement += abs(head - requests[i]);
      head = requests[i];
      visited[i]=1;
      printf(" - %d",head);

     count--;
     if(count <= 0)
        break;
     i+= direction;
    }
    printf("\nTotal head movement : %d\n",total_movement);
}

int main()
{
```

```
   int requests[MAX];
   int nreqs , head , direction,i,n;

   printf("Enter the number of requests :");
   scanf("%d",&nreqs);

   accept_requests(requests, nreqs);


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

   printf("Enter direction (-1 for left , 1 for right) :");
   scanf("%d",&direction);

   scan(head,direction,requests,nreqs);
}
```

CSCAN

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

#define MAX 200

void sort(int arr[] , int n)
{
   int i, j, temp;
   for(i=0; i<n-1 ; i++)
   {
      for(j=0; j<n-i-1 ; j++)
      {
         if(arr[j] > arr[j+1])
         {
           temp = arr[j];
           arr[j] = arr[j+1];
           arr[j+1] = temp;
         }
      }
   }
}

void cscan(int head , int direction , int requests[] , int nreqs)
{
    int totalMovement = 0;
    int count = nreqs;
    int visited[MAX] = {0};
    int i,traversed,j;

    printf("Sequence : %d->",head);
    if(direction == 1)
    {
       i=0;
```

```
   while(i<nreqs && requests[i] < head)
   {
      i++;
   }

   for(j=i ; j<nreqs ; j++)
   {
     totalMovement += abs(head - requests[j]);
      head = requests[j];
      printf("%d->",head);
   }

  totalMovement += (MAX -1-head);
   printf("%d 0",MAX-1);
  totalMovement += (MAX-1);
   head = 0;

   for(j=0 ; j<i ; j++)
   {
     totalMovement += abs(head - requests[j]);
      head = requests[j];
      printf("->%d",head);
   }
}
else
{
    if(direction == -1)
    {
        i = nreqs -1;
        while(i>0  && requests[i] > head)
        {
           i--;
        }

        for(j=i ; j>=0; j--)
        {
          totalMovement += abs(head - requests[j]);
           head = requests[j];
           printf("%d->",head);
        }

       totalMovement += head;
        printf("0 %d ",MAX-1);
       totalMovement += (MAX-1);
        head = MAX-1;

       for(j=nreqs-1 ; j>i ; j--)
       {
         totalMovement += abs(head - requests[j]);
           head = requests[j];
           printf("->%d",head);
       }
    }
}
printf("\nTotal head movement : %d\n",totalMovement);
```

```
}

int main()
{
   int requests[MAX];
   int nreqs , head , direction,i;

   printf("Enter the number of requests :");
   scanf("%d",&nreqs);

   printf("Enter requests : ");
   for(i = 0 ; i< nreqs ; i++)
   {
      scanf("%d",&requests[i]);
   }
   sort(requests,nreqs);

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

   printf("Enter direction (-1 for left , 1 for right) :");
   scanf("%d",&direction);

   cscan(head,direction,requests,nreqs);
}
```