EXPERIMENT 12
DISK SCHEDULING ALGORITHMS

AIM
write C program to disk scheduling algorithms FCFS, SCAN & CSCAN.

FCFS ALGORITHM
STEP 0: START.
STEP 1: Declare required variables
n - no of the queue
q - array to store queue elements
i - loop counter
diff - to calculate seek distance
seek - store total seek distance.
STEP 2: Prompt the user to enter the no of queue and store it in 'n'.
STEP 3: Prompt the user to enter the queue elements one by one and store them array q[i].
STEP 4: Prompt the user to enter the initial head position and store it in 'q[0]'.
STEP 5: Iterate through each queue element and calculate absolute difference between consecutive elements to determine the seek distance.
i) Update total seek distance 'seek' by adding calculated difference.

STEP 6: Display the movement from one position to another along with the seek distance for each step.
STEP 7: Display total seek distance after traversing entire queue.
STEP 8: Calculate average seek distance by dividing the total seek distance by no. of elements in queue.
STEP 9: Display average seek distance.
STEP 10: END.

SCAN ALGORITHM
STEP 0: START
STEP 1: Declare the variables named 'q' to store queue elements.
'n' - size of queue, 'i' - loop counter, 'seek' - to store total seek distance, 'cur' - current head position, 'prev' for previous head position, 'j' - loop counter, 'm' - queue size, 'cyl' - number of cylinders and 'loc' for location of the current head position.
STEP 2: Prompt the user to enter the number of cylinders and store it in 'cyl'
STEP 3: Prompt user to enter queue size and store it in variable m.
Update n = m+1.
STEP 4: Prompt user to enter queue elements one by one and store in q[i].

STEP 5: Prompt user to enter the current head position as well as previous head position and store it in cur and prev respectively.
set q[0] = cur.
STEP 6: Sort the queue elements using Bubble sort Algorithm.
STEP 7: Display sorted queue elements.
STEP 8: Find location of current head position in the sorted queue.
STEP 9: Determine seek distance based on whether current head position is before/after previous head position
i) If current head position < previous head position, then calculate seek distance by scanning towards left and then towards right
ii) If current head position > previous head position, then calculate seek distance by scanning towards right and then towards left.
iii) update total seek distance 'seek'.
STEP 10: Display total seek distance 'seek'.
STEP 11: Average seek Distance
= total seek Distance / No of elements
STEP 12: Display Average seek distance
STEP 13: END.

CSCAN ALGORITHM

STEP 0: START

STEP 1: Declare variables required. q - array to store queue elements; n - size of queue, 'seek' - store the total seek distance, 'i' & 'j' - loop counters, 'cur' - current head position 'prev' - previous head position.. 'm' - size of queue 'cyl' - number of cylinders and 'loc' for location of current head position.

STEP 2: Prompt the user to enter the number of cylinders and store it in 'cyl'.

STEP 3: Display the range of cylinders from 0 to cyl-1

STEP 4: Prompt the user to enter the queue size and store it in m. Update n=m+1

STEP 5: Prompt the user to enter the queue elements one by one and store them in array q[i].

STEP 6: Prompt the user to enter the current head position and store it in 'cur'. Set q[0] = cur.

STEP 7: Prompt the user to enter the previous head position and store it in 'prev'.

STEP 8: Store the elements after they are done sorting using Bubble sort

STEP 9: Display the sorted queue elements
STEP 10: Find location of current head position in sorted queue.
STEP 11: Determine the seek distance based on whether the current head position is before/after previous head position:
i) If current head position < previous head position, then calculate the seek distance by scanning towards the left and then restarting at the right end.
ii) If current head position > previous head position, then calculate the seek distance by scanning towards the right and then restarting at the left end.
iii) Update the total seek distance 'seek'.
STEP 12: Display Total Seek Distance 'seek'
STEP 13: Average seek Distance
= Total seek Distance / No of elements+1
STEP 14: Display the average seek distance.
STEP 15: END.

RESULT
Experiment successfully executed and output obtained.