//Program 10. Develop a C program to simulate SCAN disk scheduling algorithm.

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

void scanDiskSchedule(int request[], int n, int head, int diskSize) {
   int seekCount = 0;
   int direction =1; // 1 for right, 0 for left

   // Sort the request array in ascending order
   for (int i = 0; i < n - 1; i++)
   {
      for (int j = 0; j < n - i - 1; j++)
      {
         if (request[j] > request[j + 1])
         {
            int temp = request[j];
            request[j] = request[j + 1];
            request[j + 1] = temp;
         }
      }
   }


   // Find the index of the head in the sorted request array
   int index;
   for (index = 0; index < n; index++)
   {
      if (request[index] >= head)
      {
         break;
      }
   }


   // Calculate seek operations
   int i;
   printf("\n\nSeek sequence : ");
   if (direction == 1)
   { // Moving to the right
      // Service requests from the head to the end of the disk
      for (i = index; i < n; i++)
      {
         seekCount += abs(head - request[i]);
         printf("%d ",request[i]);
         head = request[i];
```

```
        }
        // Move to the end of the disk (if necessary) and then to the beginning
        if (head < diskSize - 1)
        {
            seekCount += abs(head - (diskSize - 1));
            head = diskSize-1;
        }
        // Service requests from the beginning to the index
        for (i = index-1; i >= 0; i--)
        {
            seekCount += abs(head - request[i]);
            printf("%d ",request[i]);
            head = request[i];
        }
    }
    else
    { // Moving to the left
        // Service requests from the head to the beginning of the disk
        for (i = index - 1; i >= 0; i--)
        {
            seekCount += abs(head - request[i]);
            printf("%d ",request[i]);

            head = request[i];
        }
        // Move to the beginning of the disk (if necessary) and then to the end
        if (head > 0)
        {
            seekCount += abs(head - 0);
            head = 0;
        }
        // Service requests from the end to the index
        for (i = index; i <=n-1; i++)
        {
            seekCount += abs(head - request[i]);
            printf("%d ",request[i]);

            head = request[i];
        }
    }
    printf("\nTotal seek count: %d\n", seekCount);
}

int main()
{
    int request[] = {53, 183, 37, 122, 14, 124, 65, 67};
    int n = sizeof(request) / sizeof(request[0]);
    int head = 53;
```

```
    int diskSize = 200; // Assume disk size is 200

    printf("Initial head position: %d\n", head);
    printf("Request queue: ");
    for (int i = 0; i < n; i++)
    {
        printf("%d ", request[i]);
    }
    printf("\n");
    scanDiskSchedule(request, n, head, diskSize);
    return 0;
}
```

Output: