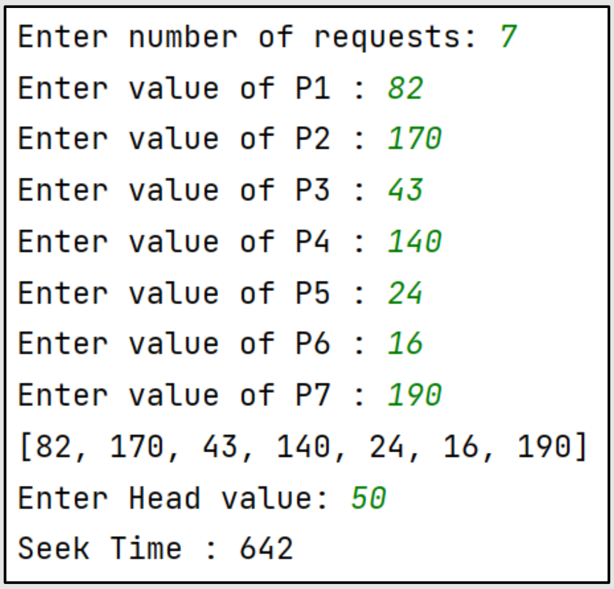
Lab 8: Disk Scheduling

1. First Come First Serve (FCFS)

```
import java.util.*;

public class FCFS {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of requests: ");
        int n = sc.nextInt();

        int[] requests = new int[n];

        for (int i=1 ; i<=n ; i++) {
            System.out.printf("Enter value of P%d : ", i);
            requests[i-1] = sc.nextInt();
        }
        System.out.println(Arrays.toString(requests));

        System.out.print("Enter Head value: ");
        int head = sc.nextInt();

        int seekTime = 0;

        for (int i=0 ; i<n ; i++) {
            if (head > requests[i]) {
                seekTime = seekTime - (requests[i] - head);
            }
            else {
                seekTime = seekTime + (requests[i] - head);
            }
            head = requests[i];
        }

        System.out.println("Seek Time : " + seekTime);

        sc.close();
    }
}
```

Output:

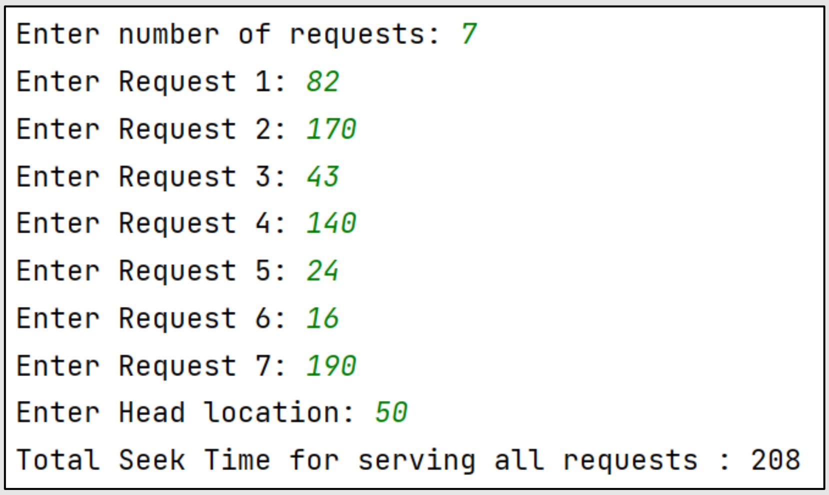


```
Enter number of requests: 7
Enter value of P1 : 82
Enter value of P2 : 170
Enter value of P3 : 43
Enter value of P4 : 140
Enter value of P5 : 24
Enter value of P6 : 16
Enter value of P7 : 190
[82, 170, 43, 140, 24, 16, 190]
Enter Head value: 50
Seek Time : 642
```

2. Shortest Seek Time First (SSTF)

```
import java.util.Scanner;

public class SSTF {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter number of requests: ");
        int n = input.nextInt();
        int[] requests = new int[n];

        for(int i = 0; i < n; i ++) {
            System.out.print("Enter Request " + (i + 1) + ": ");
            requests[i] = input.nextInt();
        }

        System.out.print("Enter Head location: ");
        int head = input.nextInt();

        int seekTime = 0;
        boolean[] completed = new boolean[n];

        // Main Programm
        for (int i=0 ; i<n ; i++) {
            int[] difference = findSeekTime(requests, head, completed);
            int index = findIndex(difference);
            seekTime += difference[index];
            completed[index] = true;
            head = requests[index];
        }

        System.out.println("Total Seek Time for serving all requests : " + seekTime);
        input.close();
    }

    public static int findDifference(int a, int b) {
        if (a > b) { return a-b; }
        else { return b-a; }
    }

    public static int[] findSeekTime(int[] requests, int head, boolean[] completed) {
        int[] difference = new int[requests.length];

        for (int i=0 ; i<requests.length ; i++) {
            if (!completed[i]) {
                difference[i] = findDifference(head, requests[i]);
            } else {
                difference[i] = Integer.MAX_VALUE;
            }
        }
        return difference;
    }

    public static int findMin (int[] array) {
        int min = Integer.MAX_VALUE;
        for (int i=0 ; i<array.length ; i++) {
            if (array[i] < min) {
                min = array[i];
            }
        }
        return min;
    }

    public static int findIndex (int[] array) {
        int i = 0;
        int index = -1;
        int min = findMin(array);
        while(i < array.length) {
            if(array[i] == min) {
                index = i;
                break;
            }
            i++;
        }
        return index;
    }
}
```

Output:

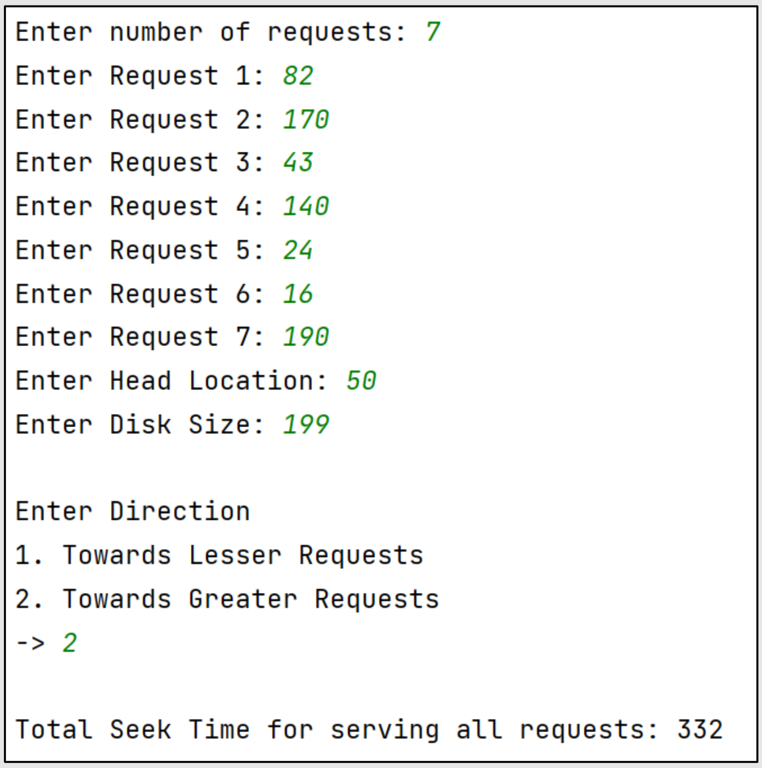


```
Enter number of requests: 7
Enter Request 1: 82
Enter Request 2: 170
Enter Request 3: 43
Enter Request 4: 140
Enter Request 5: 24
Enter Request 6: 16
Enter Request 7: 190
Enter Head location: 50
Total Seek Time for serving all requests : 208
```

3. SCAN / Elevator Algorithm

```
import java.util.Scanner;

public class SCAN {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of requests: ");
        int n = sc.nextInt();
        int[] requests = new int[n];

        for(int i = 0; i < n; i ++) {
            System.out.print("Enter Request " + (i + 1) + ": ");
            requests[i] = sc.nextInt();
        }

        System.out.print("Enter Head Location: ");
        int head = sc.nextInt();

        System.out.print("Enter Disk Size: ");
        int diskSize = sc.nextInt();

        System.out.print("\nEnter Direction\n1. Towards Lesser Requests\n2. Towards Greater Requests\n->
");
        int direction = sc.nextInt();
        int seekTime = 0;
        boolean[] completed = new boolean[n];
        int Distance = 0;

        if (direction == 1) {
            while(head >= 0) {
                for(int i = 0; i < n; i ++) {
                    if(requests[i] == head && completed[i] == false) {
                        seekTime = seekTime + Distance;
                        Distance = 0;
                    }
                }
                Distance ++;
                head --;
            }
            while(head < diskSize) {
                for(int i = 0; i < n; i ++) {
                    if(requests[i] == head) {
                        seekTime = seekTime + Distance;
                        completed[i] = true;
                        Distance = 0;
                    }
                }
                Distance ++;
                head ++;
            }
        }

        else if (direction == 2) {
            while(head < diskSize) {
                for(int i = 0; i < n; i ++) {
                    if(requests[i] == head) {
                        seekTime = seekTime + Distance;
                        completed[i] = true;
                        Distance = 0;
                    }
                }
                Distance ++;
                head ++;
            }
            while(head >= 0) {
                for(int i = 0; i < n; i ++) {
                    if(requests[i] == head && completed[i] == false) {
                        seekTime = seekTime + Distance;
                        Distance = 0;
                    }
                }
                Distance ++;
                head --;
            }
        }
        System.out.println("\nTotal Seek Time for serving all requests: " + seekTime);
        sc.close();
    }
}
```

Output:

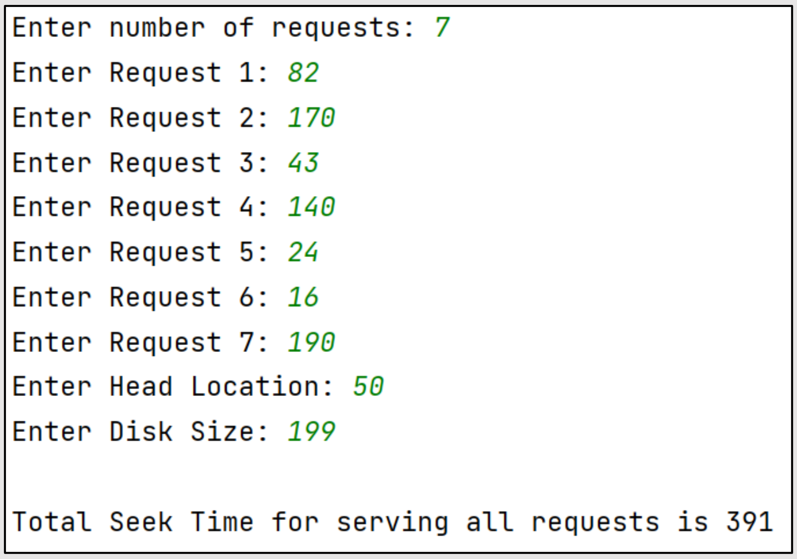


```
Enter number of requests: 7
Enter Request 1: 82
Enter Request 2: 170
Enter Request 3: 43
Enter Request 4: 140
Enter Request 5: 24
Enter Request 6: 16
Enter Request 7: 190
Enter Head Location: 50
Enter Disk Size: 199

Enter Direction
1. Towards Lesser Requests
2. Towards Greater Requests
-> 2

Total Seek Time for serving all requests: 332
```

4. Circular - SCAN (CSCAN)

```
import java.util.Scanner;

public class CSCAN {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of requests: ");
        int n = sc.nextInt();
        int[] requests = new int[n];

        for(int i = 0; i < n; i ++) {
            System.out.print("Enter Request " + (i + 1) + ": ");
            requests[i] = sc.nextInt();
        }

        System.out.print("Enter Head Location: ");
        int head = sc.nextInt();

        System.out.print("Enter Disk Size: ");
        int diskSize = sc.nextInt();

        int initialHead = head;
        int seekTime = 0;
        int Distance = 0;

        while(head < diskSize) {
            for(int i = 0; i < n; i ++) {
                if(requests[i] == head) {
                    seekTime = seekTime + Distance;
                    Distance = 0;
                }
            }
            Distance ++;
            head ++;
        }

        seekTime = seekTime + diskSize;
        head = 0;

        while(head <= initialHead) {
            for(int i = 0; i < n; i ++) {
                if(requests[i] == head) {
                    seekTime = seekTime + Distance;
                    Distance = 0;
                }
            }
            Distance ++;
            head ++;
        }
        System.out.println("\nTotal Seek Time for serving all requests is " + seekTime);
        sc.close();
    }
}
```

Output:

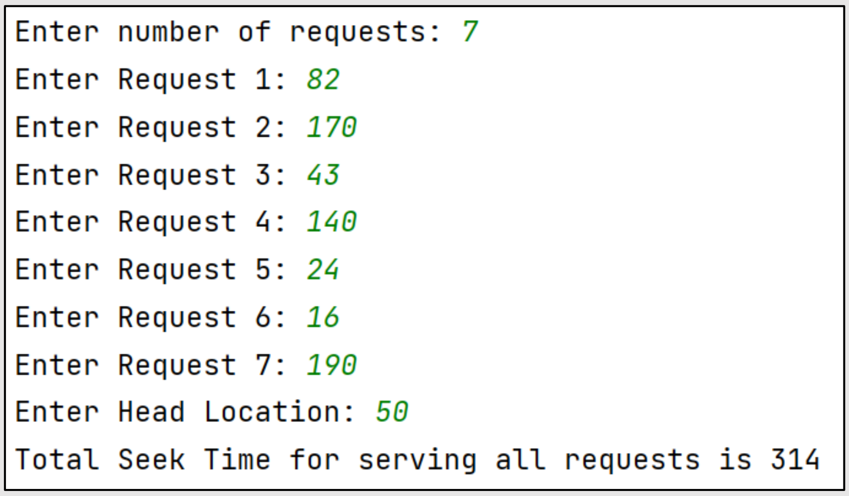


```
Enter number of requests: 7
Enter Request 1: 82
Enter Request 2: 170
Enter Request 3: 43
Enter Request 4: 140
Enter Request 5: 24
Enter Request 6: 16
Enter Request 7: 190
Enter Head Location: 50
Enter Disk Size: 199

Total Seek Time for serving all requests is 391
```

5. LOOK

```
import java.util.Scanner;

public class LOOK {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of requests: ");
        int n = sc.nextInt();
        int[] requests = new int[n];

        for(int i = 0; i < n; i ++) {
            System.out.print("Enter Request " + (i + 1) + ": ");
            requests[i] = sc.nextInt();
        }

        System.out.print("Enter Head Location: ");
        int head = sc.nextInt();
        int seekTime = 0;
        boolean[] completed = new boolean[n];
        int Distance = 0;
        int upperBound = Integer.MIN_VALUE;
        int lowerBound = Integer.MAX_VALUE;

        for(int i = 0; i < n; i ++) {
            if(requests[i] > upperBound) {
                upperBound = requests[i];
            }
            if(requests[i] < lowerBound) {
                lowerBound = requests[i];
            }
        }

        while(head < upperBound) {
            for(int i = 0; i < n; i ++) {
                if(requests[i] == head) {
                    seekTime = seekTime + Distance;
                    completed[i] = true;
                    Distance = 0;
                }
            }
            Distance ++;
            head ++;
        }

        while(head >= lowerBound) {
            for(int i = 0; i < n; i ++) {
                if(requests[i] == head && completed[i] == false) {
                    seekTime = seekTime + Distance;
                    Distance = 0;
                }
            }
            Distance ++;
            head --;
        }
        System.out.println("Total Seek Time for serving all requests is " + seekTime);
        sc.close();
    }
}
```

Output:

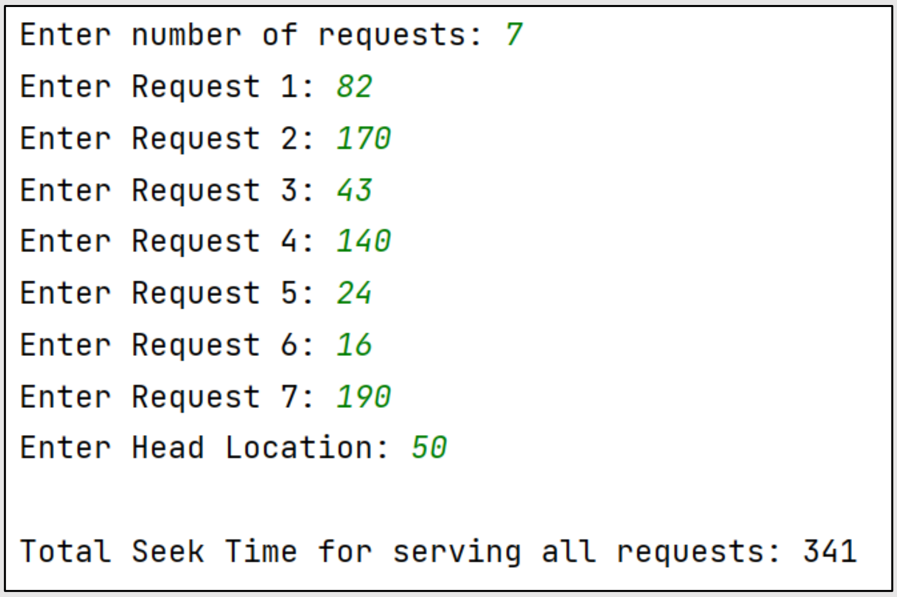


```
Enter number of requests: 7
Enter Request 1: 82
Enter Request 2: 170
Enter Request 3: 43
Enter Request 4: 140
Enter Request 5: 24
Enter Request 6: 16
Enter Request 7: 190
Enter Head Location: 50
Total Seek Time for serving all requests is 314
```

6. Circular – LOOK (CLOOK)

```
import java.util.Scanner;

public class CLOOK {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of requests: ");
        int n = sc.nextInt();
        int[] requests = new int[n];

        for(int i = 0; i < n; i ++) {
            System.out.print("Enter Request " + (i + 1) + ": ");
            requests[i] = sc.nextInt();
        }

        System.out.print("Enter Head Location: ");
        int head = sc.nextInt();
        int seekTime = 0;
        int Distance = 0;
        int upperBound = Integer.MIN_VALUE;
        int lowerBound = Integer.MAX_VALUE;
        int initialHead = head;

        for(int i = 0; i < n; i ++) {
            if(requests[i] > upperBound) {
                upperBound = requests[i];
            }
            if(requests[i] < lowerBound) {
                lowerBound = requests[i];
            }
        }

        while(head < upperBound) {
            for(int i = 0; i < n; i ++) {
                if(requests[i] == head) {
                    seekTime = seekTime + Distance;
                    Distance = 0;
                }
            }
            Distance ++;
            head ++;
        }
        head = lowerBound;


        seekTime = seekTime + upperBound - lowerBound;

        while(head < initialHead) {
            for(int i = 0; i < n; i ++) {
                if(requests[i] == head) {
                    seekTime = seekTime + Distance;
                    Distance = 0;
                }
            }
            Distance ++;
            head ++;
        }
        System.out.println("\nTotal Seek Time for serving all requests: " + seekTime);
        sc.close();
    }
}
```

Output:



```
Enter number of requests: 7
Enter Request 1: 82
Enter Request 2: 170
Enter Request 3: 43
Enter Request 4: 140
Enter Request 5: 24
Enter Request 6: 16
Enter Request 7: 190
Enter Head Location: 50


Total Seek Time for serving all requests: 341
```