Mass-Storage Structure

This covers a retired Java programming problem.

Programming Problems

10.1 Write a Java program that simulates the disk-scheduling algorithms discussed in Section 10.4. In particular, design separate classes that implement the following scheduling algorithms:

a. FCFS

b. SSTF

c. SCAN

d. C-SCAN

e. LOOK

Each algorithm will implement the following interface:

```
public interface DiskScheduler
{
   // service the requests
   // return the amount of head movement
   // for the particular algorithm
   public int serviceRequests();
}
```

The `serviceRequests()` method will return the amount of head movement required by the disk-scheduling algorithm.

Reference strings consisting of request for disk cylinders will be provided by the `Generator` class, which is available online (www.os-book.com) The `Generator` class produces random requests for cylinders numbered between 0 and 99. The API for the `Generator` class appears as follows:

```
// produce a default-sized list of cylinder requests
public Generator()
// produce a list of cylinder requests of size count
public Generator(int count)
// return the list of cylinder requests
public int[] getCylinders()
```

Each algorithm implementing the `DiskScheduler` interface must supply a constructor that is passed (1) an integer array of cylinder requests and (2) the initial cylinder position of the disk head. Assuming the `FCFS` class implements `DiskScheduler` according to the FCFS policy, an example illustrating its usage is shown below:

```
Generator ref = new Generator(1000);

int[] referenceString = ref.getCylinders();
DiskScheduler fcfs = new FCFS(referenceString, 13);
System.out.println("FCFS = " + fcfs.serviceRequests());
```

This example constructs 1,000 random cylinder requests and begins the FCFS algorithm at cylinder 13.

When you have finished, compare the amounts of head movement required by the various disk-scheduling algorithms.