Disk Scheduling Lab

This lab project addresses the implementation of IO-scheduling algorithms in an operating system.

Each IO request in an operating system is managed by using a data structure called the Request Control Block (RCB). An RCB contains the request ID, arrival timestamp, cylinder, address, and the ID of the process that posted the request. The RCB struct is defined as follows and is accessible from the rest of the code in this lab project by including the header file oslabs.h in your source code files:

```
struct RCB {
    int request_id;
    int arrival_timestamp;
    int cylinder;
    int address;
    int process_id;
}
```

The set of IO requests in the operating system that are to be processed are tracked in an IO Request Queue. This data structure is an array of RCBs of the requests.

The NULLRCB is defined as [RID:0, AT:0, CYL:0, ADDR:0, PID:0]

To determine the schedule of servicing the IO requests in an operating system, we consider three policies:

1. First-Come-First-Served Scheduling (FCFS)
2. Shortest-Seek-Time-First Scheduling (SSTF)
3. LOOK Scheduling (LOOK)

In order to implement the above policies, we need to develop methods that handle the arrival of IO requests and the completion of IO requests. That is, when a new IO request arrives, we need to figure out whether to service it immediately or to store it in the IO Request Queue. Whenever an IO request is completed, we need to figure out the next request from the IO Request Queue that needs to be serviced. The details of these methods are described below in the specification and you need to develop code for these methods that follow the specification and place the code in a file named **disk.c**. You should include the oslabs.h file.

handle_request_arrival_fcfs

This method implements the logic to handle the arrival of a new IO request in a First-Come-First-Served Scheduler. Specifically, it takes five inputs:

1. the request queue (an array of RCB structs)
2. The number of items in the request queue
3. the RCB of the currently-serviced request
4. the RCB of the newly-arriving request
5. the current timestamp.

The method returns the RCB of the newly-arriving request if the disk is free (indicated by the third parameter being a NULLRCB), otherwise, it returns the RCB of the currently-serviced request after adding the newly-arriving request to the request queue.

The signature of the method is as follows:

```
struct RCB handle_request_arrival_fcfs(struct RCB request_queue[QUEUEMAX], int *queue_cnt, struct RCB current_request, struct RCB new_request, int timestamp);
```

A sample execution input and output:

input/output	parameter	value
input	request_queue	EMPTY
input	queue_cnt	0
input	current_request	[RID:51, AT:1, CYL:53, ADDR:53, PID:51]
input	new_request	[RID:52, AT:2, CYL:54, ADDR:54, PID:52]
input	timestamp	2
output	request_queue	[RID:52, AT:2, CYL:54, ADDR:54, PID:52]
output	queue_cnt	1
output	RCB	[RID:51, AT:1, CYL:53, ADDR:53, PID:51]

Please refer to Section 5.4.3 of the Modern Operating Systems book for a detailed discussion of the First-Come-First-Served algorithm.

handle_request_completion_fcfs

This method implements the logic to handle the completion of servicing an IO request in a First-Come-First-Served Scheduler. Specifically, it takes a single input:

1. the request queue (an array of RCB structs).
2. The number of items in the request queue

The method determines the request to service next and returns its RCB.

If the request queue is empty, the method returns a NULLRCB, indicating that there is no request to service next. Otherwise, the method finds the RCB in the request queue that has the earliest arrival time. It then removes this RCB from the request queue and returns it.

The signature of the method is as follows:

```
struct RCB handle_request_completion_fcfs(struct RCB request_queue[QUEUEMAX],int *queue_cnt);
```

A sample execution input and output:

input/output	parameter	value
input	request_queue	[RID:1, AT:10, CYL:124323, ADDR:124323, PID:1]
input	queue_cnt	1
output	request_queue	EMPTY
output	queue_cnt	0
output	RCB	[RID:1, AT:10, CYL:124323, ADDR:124323, PID:1]

Please refer to Section 5.4.3 of the Modern Operating Systems book for a detailed discussion of the First-Come-First-Served algorithm.

handle_request_arrival_sstf

This method implements the logic to handle the arrival of a new IO request in a Shortest-Seek-First (also known as Shortest-Seek-Time-First) Scheduler. Specifically, it takes five inputs:

1. the request queue (an array of RCB structs)
2. The number of items in the request queue
3. the RCB of the currently-serviced request
4. the RCB of the newly-arriving request
5. the current timestamp.

The method returns the RCB of the newly-arriving request if the disk is free (indicated by the third parameter being NULLRCB), otherwise, it returns the RCB of the currently-serviced request after adding the newly-arriving request to the request queue.

The signature of the method is as follows:

```
struct RCB handle_request_arrival_sstf(struct RCB request_queue[QUEUEMAX],int
*queue_cnt, struct RCB current_request, struct RCB new_request, int timestamp);
```

A sample execution input and output:

input/output	parameter	value
input	request_queue	EMPTY
input	queue_cnt	0
input	current_request	[RID:51, AT:1, CYL:53, ADDR:53, PID:51]
input	new_request	[RID:52, AT:2, CYL:54, ADDR:54, PID:52]
input	timestamp	2
output	request_queue	[RID:52, AT:2, CYL:54, ADDR:54, PID:52]
output	queue_cnt	1
output	RCB	[RID:51, AT:1, CYL:53, ADDR:53, PID:51]

Please refer to Section 5.4.3 of the Modern Operating Systems book for a detailed discussion of the Shortest-Seek-First algorithm.

Handle_request_completion_sstf

This method implements the logic to handle the completion of servicing an IO request in a Shortest-Seek-Time-First Scheduler. Specifically, it takes three inputs:

1. the request queue (an array of RCB structs)
2. The number of items in the request queue
3. the current cylinder.

The method determines the request to service next and returns its RCB.

If the request queue is empty, the method returns NULLRCB, indicating that there is no request to service next. Otherwise, the method finds the RCB in the request queue whose cylinder is closest to the current cylinder. If there are multiple requests with the closest cylinder, then the method picks the request among these that has the earliest arrival_timestamp. The method then removes the RCB of the selected request from the request queue and returns it.

The signature of the method is as follows:

```
struct RCB handle_request_completion_sstf(struct RCB request_queue[QUEUEMAX],int
*queue_cnt,int current_cylinder);
```

A sample execution input and output:

input/output	parameter	value
input	request_queue	[RID:1, AT:72, CYL:45, ADDR:45, PID:1], [RID:2, AT:71, CYL:47, ADDR:47, PID:2], [RID:3, AT:73, CYL:43, ADDR:43, PID:3]
input	queue_cnt	3
input	current_cylinder	48
output	request_queue	[RID:1, AT:72, CYL:45, ADDR:45, PID:1], [RID:3, AT:73, CYL:43, ADDR:43, PID:3]
output	queue_cnt	2

output	RCB	[RID:2, AT:71, CYL:47, ADDR:47, PID:2]

Please refer to Section 5.4.3 of the Modern Operating Systems book for a detailed discussion of the Shortest-Seek-First algorithm.

Handle_request_arrival_look

This method implements the logic to handle the arrival of a new IO request in a LOOK (also known as Elevator) Scheduler. Specifically, it takes five inputs:

1. the request queue (an array of RCB structs)
2. The number of items in the request queue
3. the RCB of the currently-serviced request
4. the RCB of the newly-arriving request
5. the current timestamp.

The method returns the RCB of the newly-arriving request if the disk is free (indicated by the third parameter being NULLRCB), otherwise, it returns the RCB of the currently-serviced request after adding the newly-arriving request to the request queue.

The signature of the method is as follows:

```
struct RCB handle_request_arrival_look(struct RCB request_queue[QUEUEMAX],int
*queue_cnt, struct RCB current_request, struct RCB new_request, int timestamp);
```

A sample execution input and output:

input/output	parameter	value
input	request_queue	EMPTY
input	queue_cnt	0
input	current_request	[RID:51, AT:1, CYL:53, ADDR:53, PID:51]
input	new_request	[RID:52, AT:2, CYL:54, ADDR:54, PID:52]
input	timestamp	2
output	request_queue	[RID:52, AT:2, CYL:54, ADDR:54, PID:52]

output	queue_cnt	1
output	RCB	[RID:51, AT:1, CYL:53, ADDR:53, PID:51]

Please refer to Section 5.4.3 of the Modern Operating Systems book for a detailed discussion of the LOOK algorithm.

Handle_request_completion_look

This method implements the logic to handle the completion of servicing an IO request in a LOOK Scheduler. Specifically, it takes four input parameters:

1. the request queue (an array of RCB structs),
2. The number of items in the request queue
3. the current cylinder
4. the scan direction.

The method determines the request to service next and returns its RCB.

If the request queue is empty, the method returns NULLRCB, indicating that there is no request to service next. Otherwise, it picks the next request to service from the request queue.

If there are requests in the queue with the same cylinder as the current cylinder, the method picks the one among these requests with the earliest arrival time. Otherwise, if the scan direction is 1 and there are requests with cylinders larger than the current cylinder, the method picks the one among these whose cylinder is closest to the current cylinder. Otherwise, if the scan direction is 1 and there are no requests with cylinders larger than the current cylinder, the method picks the request whose cylinder is closest to the current cylinder. Otherwise, if the scan direction is 0 and there are requests with cylinders smaller than the current cylinder, the method picks the one among these whose cylinder is closest to the current cylinder. Otherwise, if the scan direction is 0 and there are requests with cylinders larger than the current cylinder, the method picks the request whose cylinder is closest to the current cylinder.

After picking the RCB from the request queue, as described above, the method removes the RCB from the queue and returns it.

The signature of the method is as follows:

```
struct RCB handle_request_completion_look(struct RCB
request_queue[QUEUEMAX],int  *queue_cnt, int current_cylinder, int scan_direction);
```

A sample execution input and output:

input/output	parameter	value
input	request_queue	[RID:1, AT:52, CYL:58, ADDR:58, PID:1], [RID:2, AT:51, CYL:58, ADDR:58, PID:2], [RID:3, AT:53, CYL:58, ADDR:58, PID:3]
input	queue_cnt	3
input	current_cylinder	58
input	scan_direction	1
output	request_queue	[RID:1, AT:52, CYL:58, ADDR:58, PID:1], [RID:3, AT:53, CYL:58, ADDR:58, PID:3]
output	queue_cnt	2
output	RCB	[RID:2, AT:51, CYL:58, ADDR:58, PID:2]

Please refer to Section 5.4.3 of the Modern Operating Systems book for a detailed discussion of the LOOK algorithm.