LAB5

Name: Aasem Mohamed Henedy

ID: 5580

Group: 4

Sec: 2

FIFO

It is the simplest form for disk scheduling where we put the head on the first disk and it continuously scans till the end of the disks but it does not provide the fastest service.

```
void FIFO(vector<int> requests, int disk_size, int head_pos, int head_dir){
   requests.insert(requests.begin(), head_pos);
   printReq(requests);
   cout<<endl;
   int head_movements = getHeadMov(requests);
   cout<<"Total head movements = "<<head_movements<<endl;
   float avg_head_mov = float(head_movements) / (requests.size()-1);
   cout<<"Average head movements = "<<avg_head_mov<<endl;
}
```

SCAN

In this method the disk arm starts from first disk and keeps scanning till the other end and servicing requests till it reaches the end then it reverses its direction and do the same operations in the opposite direction.

```
void SCAN(vector<int> requests, int disk_size, int head_pos, int head_dir){
   vector<int>::iterator it;
   if(head_dir == 1)
      sort(requests.begin(),requests.end());
   else
      sort(requests.begin(),requests.end(),greater<>());
   it = find(requests.begin(), requests.end(), head_pos);
   vector<int> req;
   if(it == requests.end()){
```

```
        req.push_back(head_pos);
        if(head_dir == 1)
            it = lower_bound(requests.begin(), requests.end(), head_pos);
        else
            it = upper_bound(requests.begin(), requests.end(), head_pos,greater<>());
    }
    int idx;
    if(it == requests.end())
        idx = requests.size()-1;
    else
        idx = it-requests.begin();

    for(int i = idx ; i < requests.size() ; i++)
        req.push_back(requests[i]);
    for(int i = idx-1 ; i >=0 ; i--)
        req.push_back(requests[i]);

    printReq(req);
    cout<<endl;
    int head_movements = getHeadMov(req);
    cout<<"Total head movements = "<<head_movements<<endl;
    float avg_head_mov = (float)head_movements / requests.size();
    cout<<"Average head movements = "<<avg_head_mov<<endl;
}
```

C-SCAN

This method is like SCAN method but the difference between them is when the disk arm reaches the other end and been reversed it does not serve any request in the opposite direction but it just pass through disks till it returns to the first end again.

```
void CSCAN(vector<int> requests, int disk_size, int head_pos, int head_dir){
   vector<int>::iterator it;
   if(head_dir == 1)
      sort(requests.begin(),requests.end());
   else
      sort(requests.begin(),requests.end(),greater<>());
   it = find(requests.begin(), requests.end(), head_pos);
   vector<int> req;
   if(it == requests.end()){
      req.push_back(head_pos);
      if(head_dir == 1)
         it = lower_bound(requests.begin(), requests.end(), head_pos);
      else
         it = upper_bound(requests.begin(), requests.end(), head_pos,greater<>());
   }
   int idx;
   if(it == requests.end())
      idx = requests.size()-1;
   else
      idx = it-requests.begin();

   for(int i = idx ; i < requests.size() ; i++)
      req.push_back(requests[i]);
   for(int i = 0 ; i < idx ; i++)
      req.push_back(requests[i]);
```

```
    printReq(req);
    cout<<endl;
    int head_movements = getHeadMov(req);
    cout<<"Total head movements = "<<head_movements<<endl;
    float avg_head_mov = (float)head_movements / requests.size();
    cout<<"Average head movements = "<<avg_head_mov<<endl;
}
```

OUTPUT