HW1 Document 資訊三乙 10727219 許志仲

1. 開發環境

平台:Windows 10

使用語言:C/C++

使用Libiary: iostream, fstream, algorithm, string, vector, queue

1. 實作方法和流程
2. 將指定文件之資料全部讀入，先讀method, timeslice，再來讀掉欄位名稱，最後重複讀完所有Process，Process包含ID, Burst, ArrivalTime, Priority四項資訊，存入自訂的名為Process的struct，struct內另外紀錄wait的時間、response ratio，以及兩個大小為5的array方便之後儲存個方法的waiting time和turnaround time，並將其Process push進vector(名為processlist)備用。
3. 依照讀到的method進行排成，method的1~5分別對應FCFS, RR, SRTF, PPRR, HRRN，6則是全部進行一次，各個排成方法會在進行中時印出Gantt chart，並且每個Process執行完之後會回去processlist紀錄waiting time和turnaround time。
4. 所有應該執行的Process執行完之後，依照ID的大小對processlist進行排序，並印出每個Process的waiting time，全部印完之後再印turnaround time。
5. 五種Method的實作方式
6. FCFS

開始時先從processlist將資料複製出來，存進名為list的Process vector，並依照arrival time做排序，實作時使用一個名為pqueue的Process queue，初始化目前工作ID及剩餘工作時間後進入迴圈做排程。

在排程迴圈中，會針對每一個時間區段(1)做處理，先將arrival的Process push進pqueue裡面等待，接下來檢查CPU目前工作是否完成，若是已經完成就將結果(waiting time, turnaround time)紀錄回processlist，並從queue裡將工作pop出來進行新的工作。一直執行直到所有工作都已經執行完為止(程式設計上為list和pqueue為空)。並在每一個單位依目前工作印出甘特圖。

1. RR

開始時先從processlist將資料複製出來，存進名為list的Process vector，並依照arrival time做排序，實作時使用一個名為pqueue的Process queue，初始化目前工作ID、剩餘工作時間以及以工作時間後進入迴圈做排程。

在排程迴圈中，會針對每一個時間區段(1)做處理，先將arrival的Process push進pqueue裡面等待，首先檢查工作時間是否已到達timeslice，如果是的話就把目前Process push回pqueue，並pop出新的Process，接下來檢查CPU目前工作是否完成，若是已經完成就將結果(waiting time, turnaround time)紀錄回processlist，並從queue裡將工作pop出來進行新的工作。一直執行直到所有工作都已經執行完為止(程式設計上為list和pqueue為空)。並在每一個單位依目前工作印出甘特圖。

1. SRTF

開始時先從processlist將資料複製出來，存進名為list的Process vector，並依照arrival time做排序，實作時使用一個名為pqueue的Process priority queue，此最小優先佇列的判斷條件為剩餘工作時間，初始化目前工作ID及剩餘工作時間後進入迴圈做排程。

在排程迴圈中，會針對每一個時間區段(1)做處理，先將arrival的Process push進pqueue裡面等待，接下來檢查目前工作的剩餘時間和pqueue的top Process的剩餘時間，只要目前工作的剩餘時間比較多就將其push回pqueue，改為執行pqueue的top Process，最後檢查CPU目前工作是否完成，若是已經完成就將結果(waiting time, turnaround time)紀錄回processlist，並從queue裡將工作pop出來進行新的工作。一直執行直到所有工作都已經執行完為止(程式設計上為list和pqueue為空)。並在每一個單位依目前工作印出甘特圖。

1. PPRR

開始時先從processlist將資料複製出來，存進名為list的Process vector，並依照arrival time做排序，實作時使用一個名為pqueue的Process vector，這裡之所以改為使用vector而不是queue是因為之後方便之後RR時較容易將Process插入正確的位置，初始化目前工作ID及剩餘工作時間後進入迴圈做排程。

在排程迴圈中，會針對每一個時間區段(1)做處理，先將arrival的Process push進pqueue裡面等待，若是執行時間到達timeslice，先檢查list中的Procss是否有相同的Priority需要進行round robin，如果有就把目前Process插入list中同樣Priority最尾端的位置，同時改為執行list中的第一個Process，接下來檢查CPU目前工作是否完成，若是已經完成就將結果(waiting time, turnaround time)紀錄回processlist，並從queue裡將工作pop出來進行新的工作。一直執行直到所有工作都已經執行完為止(程式設計上為list和pqueue為空)。並在每一個單位依目前工作印出甘特圖。

1. HRRN

開始時先從processlist將資料複製出來，存進名為list的Process vector，並依照arrival time做排序，實作時使用一個名為pqueue的Process vector，這裡之所以改為使用vector而不是queue是因為之後方便更新response ratio，初始化目前工作ID及剩餘工作時間後進入迴圈做排程。

在排程迴圈中，會針對每一個時間區段(1)做處理，先將arrival的Process push進pqueue裡面等待，並且更新所有Process的response ratio，並且依照更新完的response ratio大小對pqueue做排序，接下來檢查CPU目前工作是否完成，若是已經完成就將結果(waiting time, turnaround time)紀錄回processlist，並從queue裡將工作pop出來進行新的工作。一直執行直到所有工作都已經執行完為止(程式設計上為list和pqueue為空)。並在每一個單位依目前工作印出甘特圖。