1. 開發環境

Windows11,Anaconda Python 3.8(需安裝程式裡面含有的 package 才能正常的執行)

2. 實作方法和流程

FCFS:利用 python dictionary 儲存資訊，對 arrival time, ID 進行排序，而就照排好的序列一個一個做完，當其他程序再等待就計算每個 waiting time，做完就設置此程序的 turnaround time。

RR: 利用 python dictionary 儲存資訊，對 arrival time, ID 進行排序，利用 queue 把在 current time 時間內的排序好的程序一個一個加進 queue，加完之後，再 pop 一個出來做一個 time slice 如果做完或 time out 先把小於 current time 的程序加入 queue，再把沒做完的程序加回 queue，假如做完就不用加回 queue 裡。當其他程序再等待就計算每個 waiting time，做完就設置此程序的 turnaround time。

SRTF:按照 CPU Burst, Arrival Time, ID 排序，每做一個時間單位就按照 CPU Burst, Arrival Time, ID 排序且更新一次，反覆直到做完，當其他程序再等待就計算每個 waiting time，做完就設置此程序的 turnaround time。

PPRR: 按照 Priority, Arrival Time, ID 排序，也使用了 queue，最前面的一定是最高優先且先做的程序，每做一個時間單位就更新一次看有無更高 Priority 的程序，當其他程序再等待就計算每個 waiting time，做完就設置此程序的 turnaround time。

HRRN:用一個 list，在做完一個程序就更新一次在時間內可執行程序的 Response ratio，按照這個大小排序，拿最大的程序出來做完，在重複更新一次，重複上述步驟直到做完，當其他程序再等待就計算每個 waiting time，做完就設置此程序的 turnaround time。

3. 不同排程法的比較

FCFS 和 RR 基本上演算法差不多，就我在寫這兩個排程的時候，發現到兩個的差異是 RR 比較公平一點，但就只是加了 time slice 的 FCFS，假如 time slice 太大基本上排程的結果就會和 FCFS 一樣，time slice 太小的話就是一直頻繁的切換 process 其實還蠻浪費效能和效率的。

SRTF 可以減少平均的等待時間，原因就是當還沒完成工作的程序越多，那等待的程序就越多，平均等待時間就越大，所以這個排程先把快結束的程序處理，優點是可以減少等待程序的數量進而減少平均等待時間且注重執行時間，缺點是沒有注意等待的時間。

PPRR 就是加上 Priority 的 RR 排程，只考慮 Priority，當前 High Priority 的排程會最先做完，除非有更高優先權的程序進來，但假設一直有更高優先度的程序一直加進來會導致很早來但優先度低的程序會使 turnaround time 越來越大，而程序有可能會餓死，在這次作業並沒有加入時間升級的機制，所以此排程是有可能讓程序餓死的狀況。

HRRN

注重執行的時間和等待的時間，優先順序會因時間的增加，使等待的程序的優先程度上升，一開始比例最高的會先做完，而做完之後算比例，再挑下一個比例高的做完，反覆地直到沒有下一個程序。

4. 結果與討論

```
All
==          FCFS==
-666669999AAAAAAAA7RRRRRRD11KKK444TTTTTT3333885555552220000
==           RR==
-69A67R9A6DR9A6R19AK6R14ATK3R84A5TK3R284A5T032A5T0325T05T05
==          SRTF==
-997996D6666R11KKK444RR88RRR3333222T0000TTTTT555555AAAAAAAA
==          PPRR==
-99996RD6R6R611R6RR4TT3333TTTT4488A500002A52A52A5A5A5AAKKK7
==          HRRN==
-6666679999DRRRRRR11AAAAAAAAKKK444883333TTTTTT2225555550000
```

Waiting Time

ID	FCFS	RR	SRTF	PPRR	HRRN
0	19	18	0	0	19
1	13	8	0	0	5
2	22	19	2	14	16
3	18	25	6	0	14
4	13	19	0	11	13
5	20	27	19	21	23
6	0	15	6	11	0
7	15	2	0	55	3
8	21	14	0	9	11
9	5	13	1	0	6
10	8	37	49	45	18
13	18	3	0	0	4
20	13	17	0	40	13
27	16	28	19	10	9
29	14	31	19	4	20

Turnaround Time

ID	FCFS	RR	SRTF	PPRR	HRRN
0	23	22	4	4	23
1	15	10	2	2	7
2	25	22	5	17	19
3	22	29	10	4	18
4	16	22	3	14	16
5	26	33	25	27	29
6	5	20	11	16	5
7	16	3	1	56	4
8	23	16	2	11	13
9	9	17	5	4	10
10	16	45	57	53	26
13	19	4	1	1	5
20	16	20	3	43	16
27	22	34	25	16	15
29	20	37	25	10	26

```
All
==        FCFS==
1111111111122333445555555
==          RR==
1112233311144555111555511
==        SRTF==
1223334455555511111111111
==        PPRR==
1111111111133344555555522
==        HRRN==
1111111111122443335555555
```

Waiting Time

ID	FCFS	RR	SRTF	PPRR	HRRN
1	0	13	13	0	0
2	10	2	0	21	10
3	10	2	0	8	12
4	11	6	1	9	8
5	11	9	1	9	11

Turnaround Time

ID	FCFS	RR	SRTF	PPRR	HRRN
1	11	24	24	11	11
2	12	4	2	23	12
3	13	5	3	11	15
4	13	8	3	11	10
5	17	15	7	15	17

```
All
==        FCFS==
1111111111111111111-----22222222222222222222222233333333333333333333333344444444444444----------55555555556666666666
==          RR==
1111111111111111111-----22222222223333333333222222222233333333332222244444444443333344444----------55555555556666666666
==        SRTF==
1111111111111111111-----22222222222222222222222233333333333333333333333344444444444444----------55555555556666666666
==        PPRR==
1111111111111111111-----22222222223333333333222222222233333444444444444444422222333333333----------55555666666666655555
==        HRRN==
1111111111111111111-----22222222222222222222222233333333333333333333333344444444444444----------55555555556666666666
```

Waiting Time

ID	FCFS	RR	SRTF	PPRR	HRRN
1	0	0	0	0	0
2	0	20	0	30	0
3	20	30	20	35	20
4	15	15	15	0	15
5	0	0	0	10	0
6	5	5	5	0	5

Turnaround Time

ID	FCFS	RR	SRTF	PPRR	HRRN
1	20	20	20	20	20
2	25	45	25	55	25
3	45	55	45	60	45
4	30	30	30	15	30
5	10	10	10	20	10
6	15	15	15	10	15

SRTF 真的減少了平均等待時間，對於 cpu burst 低的 turnaround time 也變小，time slice 很小對於 RR 並沒有太大的幫助，必須一直 context switch，假設程序能在 time slice 裡做完會比較接近 FCFS，PPRR 算是針對 RR 做一個優先權的排序，高優先度的會比較先做完，HRRN 的結果有點類似 FCFS 或是 RR，都是多出一種 INDEX 做排序的機制，HRRN 多出了 Response Ratio 的排序機制，也就是 High response ratio first，比較可以將 waiting 比較久時間的程序把 waiting 時間分布給那些沒有等很久的程序。