

# 網路程式設計實務 期末專題

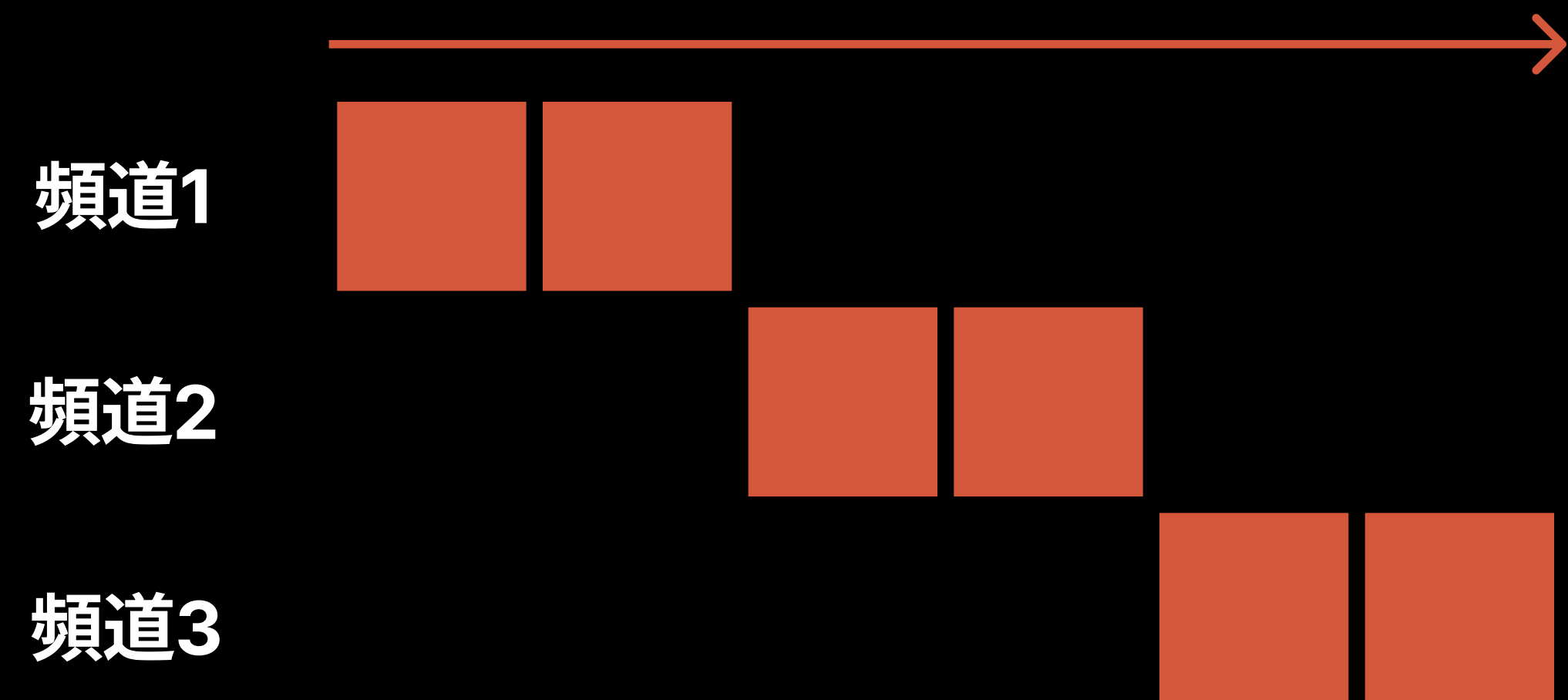
姓名：蘇泓叡

學號：C112156233

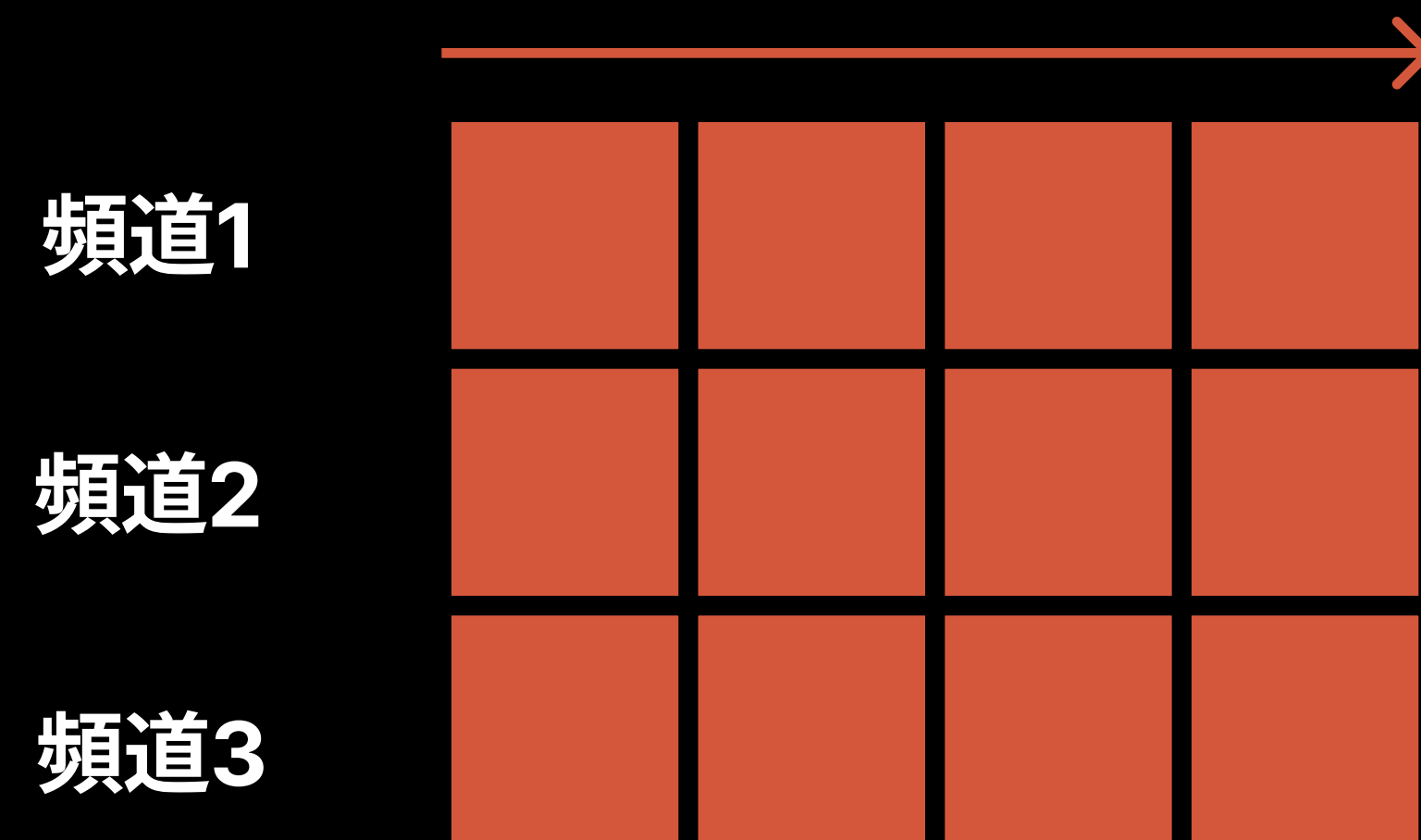
# 背景與目標

## Wi-Fi 7 MLO (Multi-Link Operation)

傳統 Wi-Fi：同一時間只能用一種頻段



Wi-Fi 7：可同時使用多個頻道傳輸



# 背景與目標

## 專題目標

用 2D 陣列實作排班表

比較 FCFS 和 Backfilling 兩種排班演  
算法

# 2D 陣列排班表

```
1 // 建立 2D 陣列
2 int** table;
3 int rows = 4;
4 int cols = 10;
5
6 // 動態配置記憶體
7 table = new int*[rows];
8 for (int i = 0; i < rows; i++) {
9     table[i] = new int[cols];
10     for (int j = 0; j < cols; j++) {
11         table[i][j] = 0; // 初始化為 0 (空閒)
12     }
13 }
14
15 // 存取方式
16 table[0][2] = 1; // 資源 0, 時段 2 分配給工作 1
17
18 // 0 = 空閒, 其他數字 = 工作 ID
```

		時段(columns)									
		0	1	2	3	4	5	6	7	8	9
資源 (rows)	0	0	0	1	1	0	0	0	3	3	3
	1	0	0	1	1	2	2	2	0	0	0
	2	0	0	0	0	2	2	2	0	0	0
	3	0	0	0	0	2	2	2	0	0	0

# 類別設計

## Class Schedule

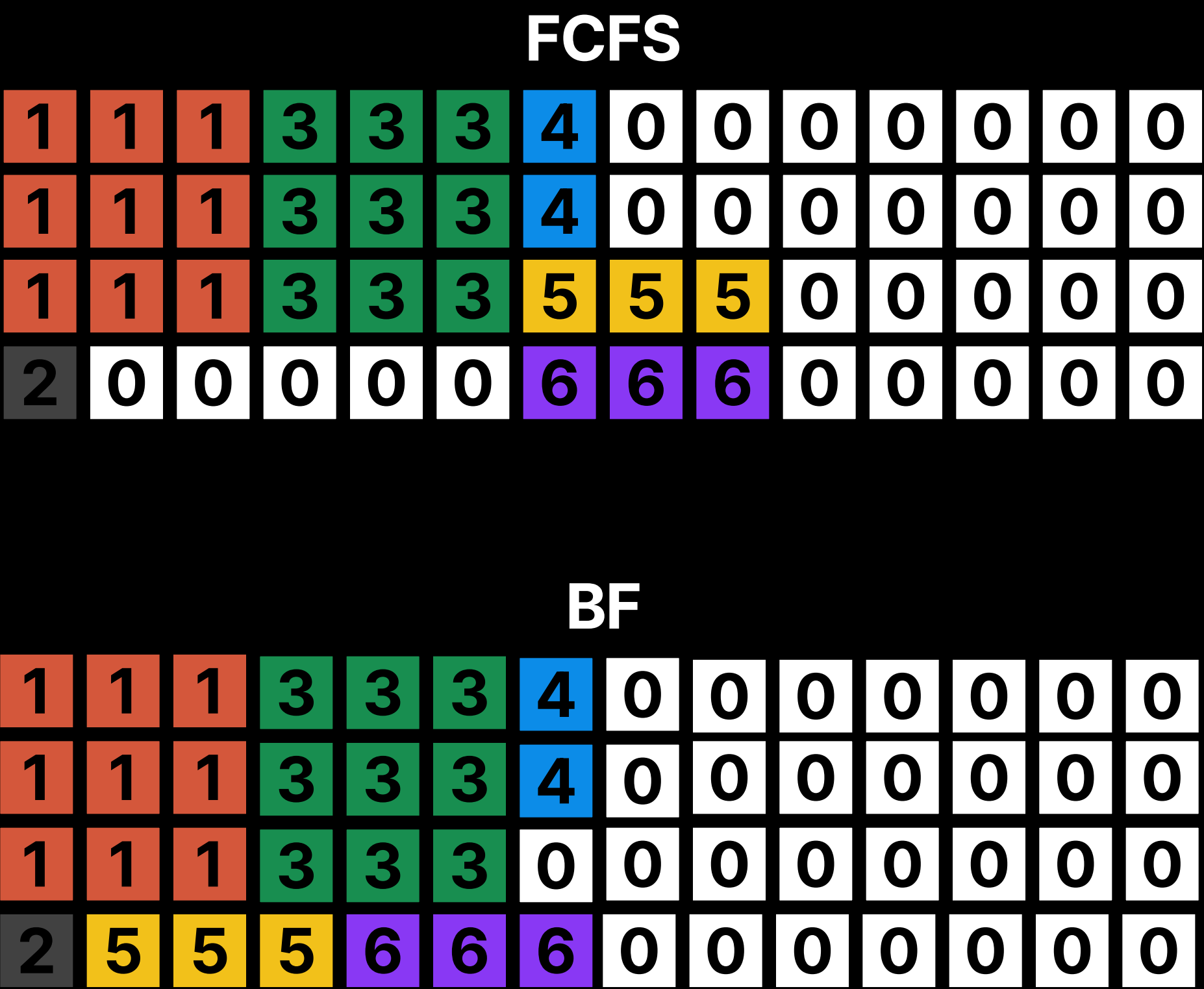
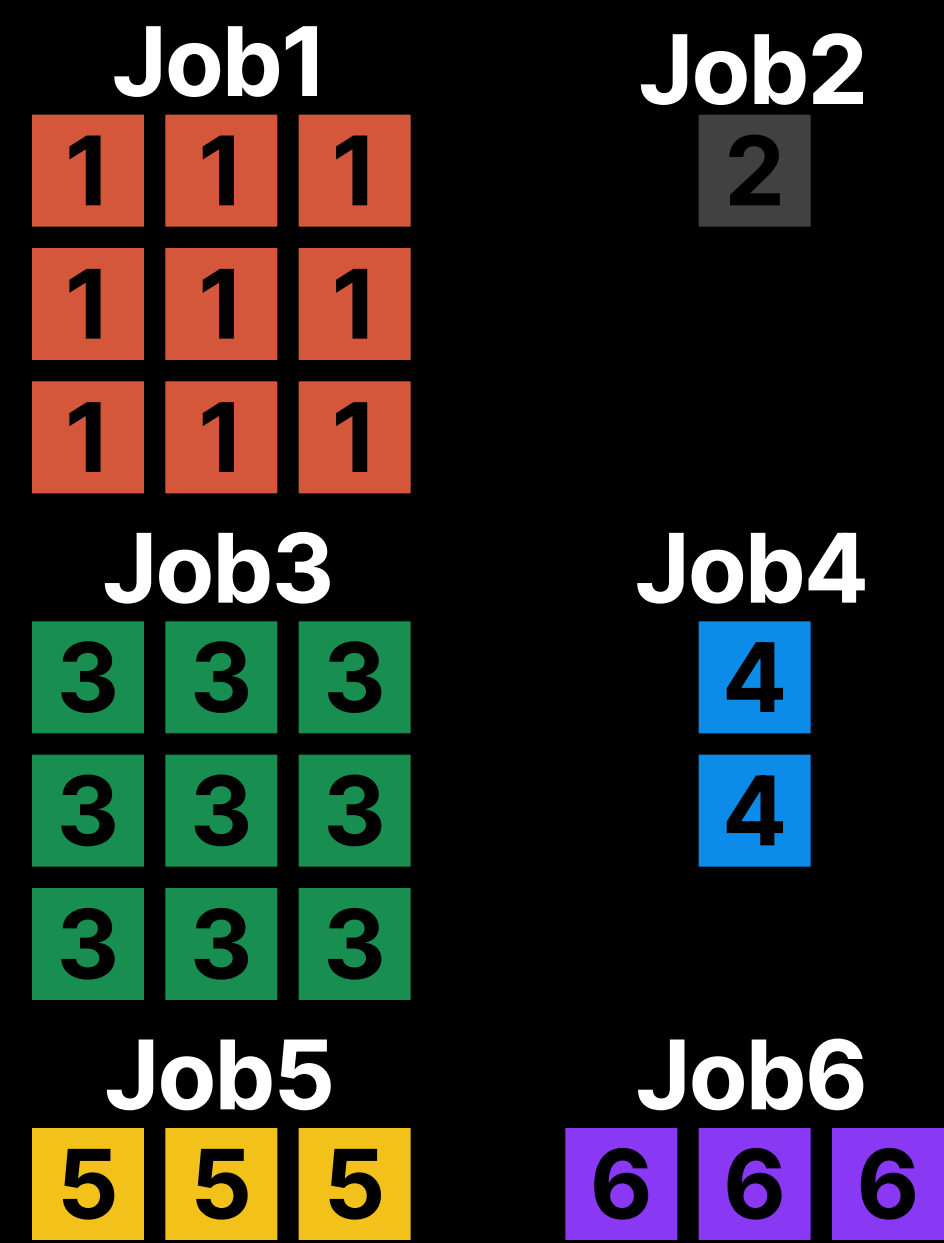
```
1  int** table;           // 2D 陣列
2  int rows, cols;       // 大小
3  int horizon;          // 排班邊界
4  int jobCount;          // 已排班數
5  int lastJobStartCol // 上一個工作開始的 col
6
7  get(), set(), test(), print()
```

## Class MLO

```
1  Schedule* fcfs; // FCFS 排班表
2  Schedule* bf;   // BF 排班表
3
4  scheduleFCFS(), scheduleBF()
```

# 排班表展示

Jobs  
(依 arrivalTime 排序)



## Throughput 比較

	FCFS	BF
已排班數	6	6
最後時段	10	8
Throughput	0.6	0.75

結論  
Backfilling 效率比 FCFS  
高約 25%