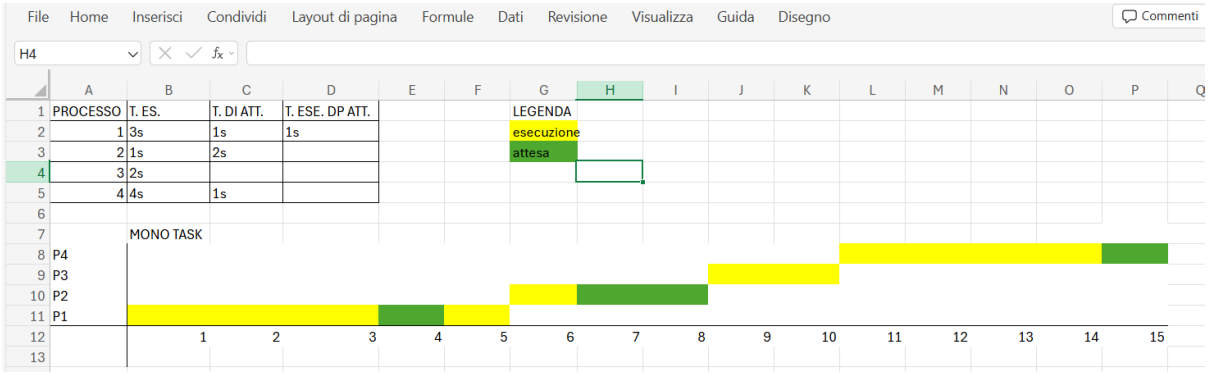
In questo esercizio cerchiamo di mostrare graficamente la gestione dei processi e la pianificazione dell'utilizzo della CPU. Vedremo i vari approcci utilizzati nel tempo, mono-tasking e multi-Tasking.

Nell'esercizio in questione abbiamo 4 processi con tempi di esecuzione e di attesa input/output diversi uno dall'altro come riportati nella tabella.

Processo	Tempo di esecuzione	Tempo di attesa	Tempo di esecuzione dopo attesa
P1	3 secondi	1 secondo	1 secondo
P2	1 secondo	2 secondi	-
P3	2 secondi	-	-
P4	4 secondi	1 secondo	-

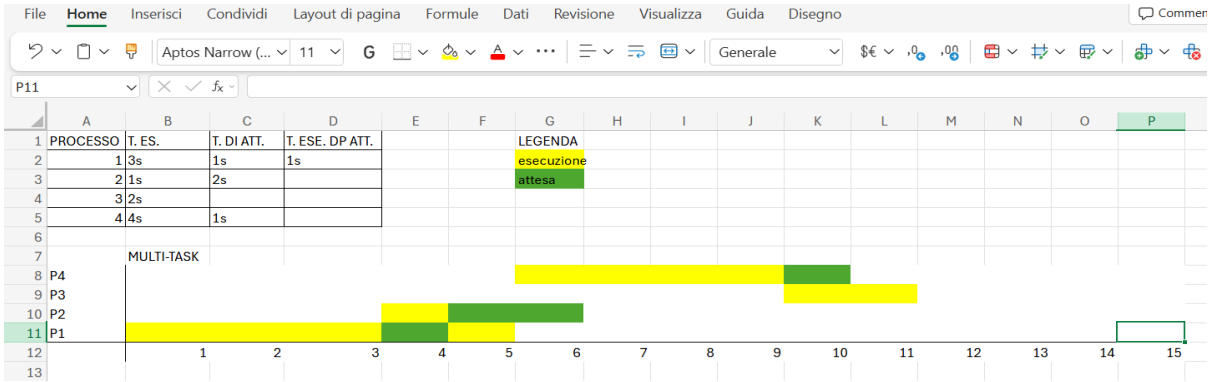
In questo caso noi andremo a considerare 3 scenari:

- Mono-tasking: Sistema che non supporta l'esecuzione parallela di più processi. I sistemi operativi mono-tasking sono sistemi poco efficienti visto che ci sono molti periodi nei quali la CPU non viene utilizzata per colpa dei tempi di attesa.



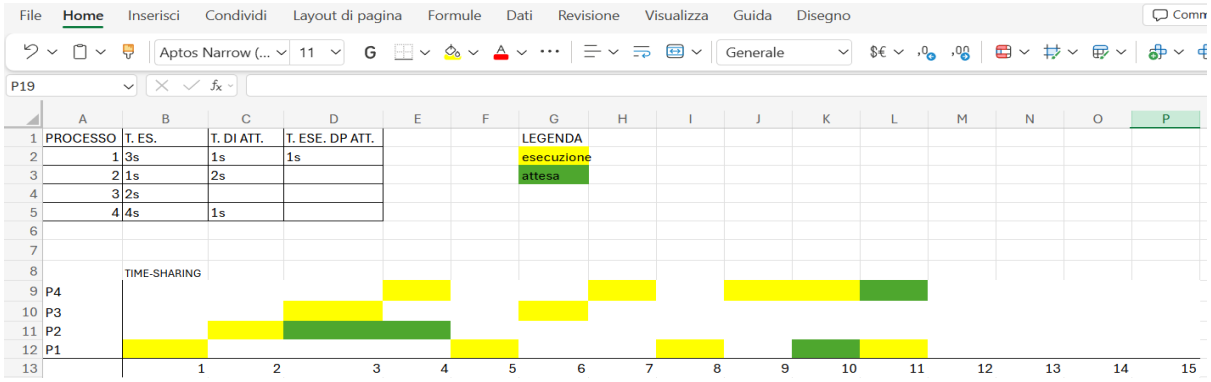
In questo processo vediamo come ogni processo inizia solo nel momento in cui il precedente è terminato. Questo comporta tempi di attesa che non vengono sfruttati per completare o avviare altri processi avendo quindi tempistiche più lunghe per completarli tutti.

- Multi-tasking: Sistemi operativi che permettono l'esecuzione contemporanea di più programmi. Lavora su sistema di prelazione che fa in modo che quando un processo è in un momento di attesa, questo periodo viene utilizzato per far avviare un altro processo



Con questo metodo si evince già da un primo sguardo come le tempistiche risultano ridotte visto che i periodi morti vengono usati per iniziare altri processi. Ad esempio, vediamo come nel primo momento di attesa (P1-4) ha inizio la prima parte del secondo processo (P2-4), quando quest'ultimo ha finito ed ha inizio il suo tempo di attesa il primo applicativo torna in esecuzione per far sì che venga completato in modo da sfruttare quel momento di attesa per completare il primo applicativo. Per poi attuare lo stesso procedimento per gli altri applicativi.

- Time-sharing In un sistema time-sharing ogni processo viene eseguito in maniera ciclica per piccole porzioni di tempo che prendono il nome di «quanti».



In questo sistema anche se apparentemente le tempistiche sono uguali a quello del multi-tasking il procedimento utilizzato per arrivare al risultato è molto diverso e questo fa sì che questo metodo di lavoro non venga utilizzato, questo perché in questo metodo ogni processo viene suddiviso in piccoli parti, che vengono eseguite in maniera ciclica indipendentemente dal tempo necessario di esecuzione e di attesa, facendo sì che gli applicativi vengono completati parzialmente ogni volta. Nel caso specifico vediamo che il primo applicativo lanciato (il P1) è anche l'ultimo che viene completato.

Sicuramente, come abbiamo dimostrato, i sistemi multi-tasking e time-sharing sono molto più efficienti rispetto al mono-tasking. Notiamo come nel multi-tasking il P1 continua la sua esecuzione ininterrottamente fino a che non cambia il suo stato in attesa cosa invece che non accade nel time-sharing dove l'esecuzione dei "quanti" è stabilita a monte.

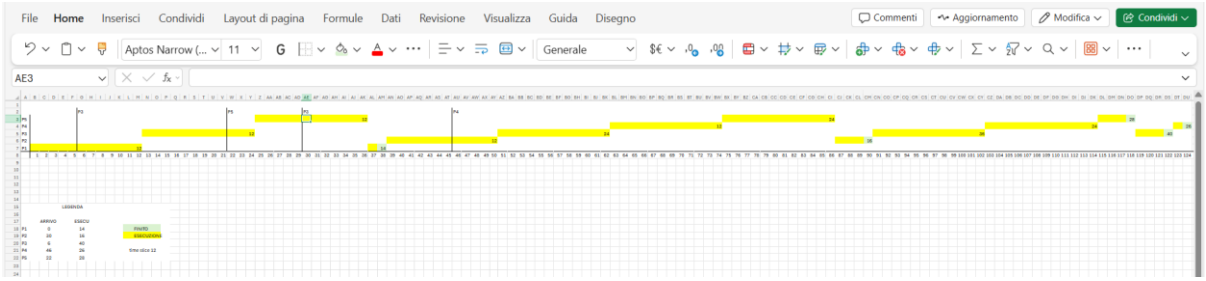
Parte facoltativa

In questa parte facoltativa dell'esercizio prendiamo in esame un insieme di 5 processi con tempi di arrivo diversi ed esecuzione diversi uno dall'altro.

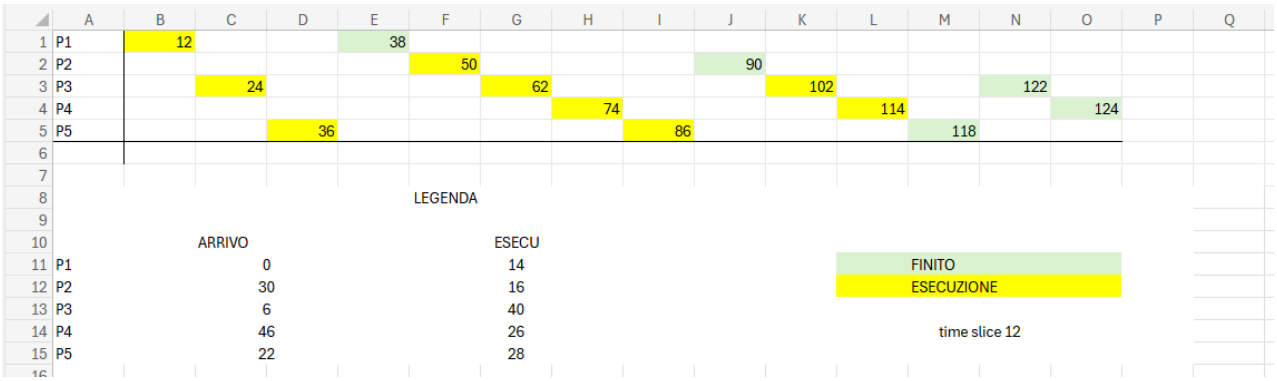
Processo	Tempo di arrivo (t_0)	Tempo di esecuzione (T_x)
P1	0	14
P2	30	16
P3	6	40
P4	46	26
P5	22	28

provando a mostrare graficamente e con l'aiuto di una tabella di descrivere lo scheduling di suddetti processi con politica di ROUND ROBIN attribuendo un timeslice di 12 millisecondi.

Prima di iniziare per Round Robin si intende un algoritmo usato dai Sistemi Operativi per assegnare la risorsa "tempo di esecuzione CPU" ai processi che competono per essa. Con questo processo i quanti sono assegnati a ciascun processo in porzioni uguali e in modo circolare, trattando tutti i processi senza priorità assegnando l'ordine in base all'arrivo. Questo processo in pratica è la fusione della politica di scheduling First In First Out (i processi sono avviati in base all'ordine di arrivo) e il concetto di timeslice (ossia quel concetto che prevede una durata di tempo limite di processo in esecuzione).



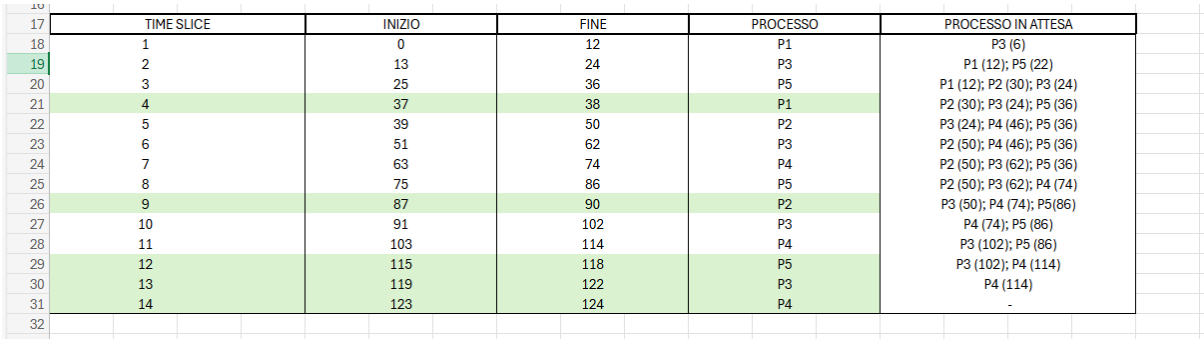
In questa rappresentazione ho cercato di mostrare graficamente con linee verticale dove un programma fosse pronto per iniziare, con le linee orizzontali di colore giallo la durata di esecuzione e con il colore verde il momento nel quale un programma avesse completato la sua esecuzione.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	P1	12			38												
2	P2					50				90							
3	P3		24				62				102			122			
4	P4							74				114			124		
5	P5			36					86				118				
6																	
7																	
8						LEGENDA											
9																	
10			ARRIVO				ESECU										
11	P1		0				14					FINITO					
12	P2		30				16					ESECUZIONE					
13	P3		6				40										
14	P4		46				26					time slice 12					
15	P5		22				28										

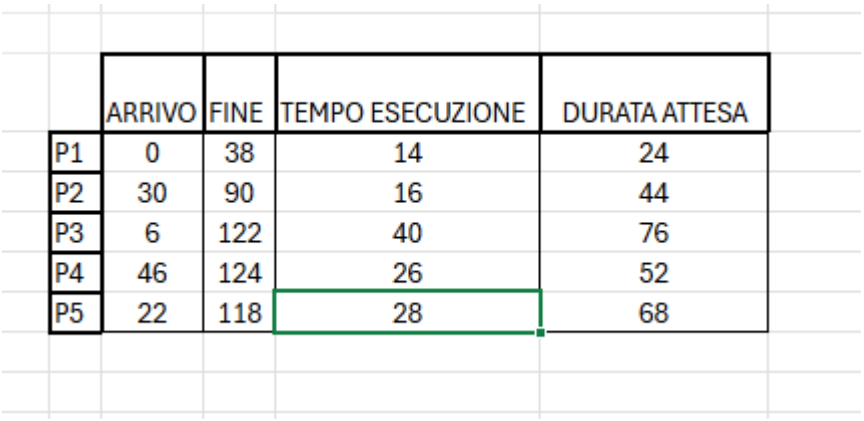
fig. 1

Nella fig.1 ho provato a mettere in evidenza i processi in esecuzione mostrando solo i millisecondi alla fine di ogni cambio di esecuzione evidenziando con il verde quando un processo fosse finito. In questo modo ho provato a rendere più intuitivo la ciclicità del processo e come questo si comporta quando uno o più programmi hanno terminato la loro esecuzione.



TIME SLICE	INIZIO	FINE	PROCESSO	PROCESSO IN ATTESA
1	0	12	P1	P3 (6)
2	13	24	P3	P1 (12); P5 (22)
3	25	36	P5	P1 (12); P2 (30); P3 (24)
4	37	38	P1	P2 (30); P3 (24); P5 (36)
5	39	50	P2	P3 (24); P4 (46); P5 (36)
6	51	62	P3	P2 (50); P4 (46); P5 (36)
7	63	74	P4	P2 (50); P3 (62); P5 (36)
8	75	86	P5	P2 (50); P3 (62); P4 (74)
9	87	90	P2	P3 (50); P4 (74); P5(86)
10	91	102	P3	P4 (74); P5 (86)
11	103	114	P4	P3 (102); P5 (86)
12	115	118	P5	P3 (102); P4 (114)
13	119	122	P3	P4 (114)
14	123	124	P4	-

fig. 2



	ARRIVO	FINE	TEMPO ESECUZIONE	DURATA ATTESA
P1	0	38	14	24
P2	30	90	16	44
P3	6	122	40	76
P4	46	124	26	52
P5	22	118	28	68

Fig. 3

Nelle ultime due rappresentazioni ho provato, tramite tabella, di mettere in evidenza cosa accade dietro a un programma in esecuzione. Nella fig.2, infatti, notiamo che per ogni processo in esecuzione ci sono processi che sono in attesa e da che momento. Nella fig.3 invece provo a mettere in evidenza la durata di attesa che ogni processo ha dal tempo di arrivo a quando termina.