(punti totali: 69/90)

+12/1/32+

Sistemi Operativi TP 2019
Prova Scritta

Nome e Cognome:

.....

Tempo a disposizione: 90 minuti / Documentazione ammessa: 1 foglio A5 manoscritto.

Domande a risposta multipla [63 punti]

Ogni domanda può avere una o più risposte corrette. Una risposta completamente corretta è valutata 3 punti.

Domanda 1 ◇ Il gestore (*handler*) di un segnale POSIX...

- [ ] viene eseguito in modo sincrono
- [ ] è richiamato non appena un segnale viene bloccato nella maschera
- [ ] permette di gestire i segnali in modo sincrono
- [x] può essere condiviso tra più segnali diversi

3/3

Domanda 2 ◇ Nello scheduling real-time il *laxity-time*...

- [x] dipende dal tempo di esecuzione di un task e dalla sua deadline
- [ ] di regola è zero nei sistemi che utilizzano gli algoritmi RMA/RMS
- [x] può essere uguale a zero
- [ ] dipende dal tempo di esecuzione di un task e dalla sua periodicità

3/3

Domanda 3 ◇ Un processo X si trova nello stato *zombie*...

- [ ] quando termina e il padre chiama **waitpid**
- [ ] quando attende che il processo padre termini l'esecuzione
- [ ] quando è pronto per essere eseguito
- [x] se termina dopo il processo padre

3/3

Domanda 4 ◇ Voglio valutare i tempi di esecuzione di un programma single-thread su un sistema multi-core. Utilizzo uno scheduling di tipo *round-robin* che permette all'utente di scegliere una *policy* tra *affinity*, *soft affinity* e *hard affinity*. Se volessi ottenere le migliori performance di esecuzione per il mio programma (i.e. il tempo minore di esecuzione)...

- [ ] la policy non ha importanza perché il programma di benchmark non è multi-thread
- [ ] scelgo la policy "nessuna *affinity*"
- [ ] scelgo la policy "*soft affinity*"
- [x] scelgo la policy "*hard affinity*"

3/3

Domanda 5 ◇ In un sistema che implementa lo scheduling a lotteria abbiamo 4 processi A,B,C e D con le seguenti priorità: A (priorità 1), B (priorità 2), C (priorità 4), D (priorità 6). La priorità minima è 1, la priorità massima è 6. Lo scheduler è pre-emptive e assegna un quanto di 10ms. Se lasciamo in esecuzione questi processi per un tempo totale complessivo di un'ora, con buona probabilità...

- [x] B avrà ottenuto la CPU per meno di 20 minuti
- [ ] A avrà ottenuto la CPU per più di 20 minuti
- [ ] C avrà ottenuto la CPU per meno di 15 minuti
- [x] D avrà ottenuto la CPU per più di 20 minuti

3/3

Domanda 6 ◇ Per implementare la *preemption*...

- [ ] assegno una priorità più bassa ai processi che devono essere interrotti periodicamente
- [x] posso utilizzare degli *interrupt*
- [ ] devo minimizzare il tempo di risposta del sistema
- [ ] ogni thread deve fare periodicamente uno *yield*

3/3

Domanda 7 ◇ L'algoritmo di scheduling sui sistemi Unix tradizionali...

- [ ] implementa una schedulazione con priorità statiche
- [ ] esegue i processi con ordine FCFS
- [x] esegue i processi con ordine Round-Robin
- [x] implementa una schedulazione con priorità dinamiche

3/3

Domanda 8 ◇ Lo scheduler del sistema operativo Windows...

- [x] determina la priorità di un processo sulla base della priorità dei thread
- [x] utilizza un sistema di priorità variabile (dinamico)
- [ ] permette all'utente di assegnare un *boost* di priorità temporaneo ai thread
- [x] determina la priorità dei thread sulla base della priorità dei processi

0/3

Domanda 9 ◇ Una *policy* di scheduling basata sulla *soft-affinity*...

- [ ] non si applica a processi/thread realtime
- [ ] obbliga lo scheduler ad assegnare un thread sempre allo stesso core
- [ ] è più efficace della *hard-affinity* per risolvere i problemi di *locality*
- [x] permette allo scheduler di cambiare il core assegnato a un thread

3/3

Domanda 10 ◇ Tramite il *fork*...

- [ ] viene assegnato un nuovo PID al processo corrente
- [x] è possibile creare un nuovo thread all'interno del processo
- [x] è possibile eseguire più programmi all'interno dello stesso processo
- [ ] viene creato un nuovo contesto di esecuzione

0/3

Domanda 11 ◇ In un sistema che implementa lo scheduling Fair Share ho 3 utenti X, Y e Z e un programma per il calcolo della meteo P con tempo di esecuzione costante T. Contemporaneamente, l'utente X mette in esecuzione 4 istanze del programma P, Y mette in esecuzione un'istanza del programma P, mentre Z mette in esecuzione 2 istanze del programma P. In una situazione del genere...

- [x] a fine esecuzione ogni istanza avrà ricevuto complessivamente lo stesso tempo CPU
- [x] a fine esecuzione ogni utente avrà ricevuto complessivamente lo stesso tempo CPU
- [ ] tutte le istanze del programma P di tutti gli utenti termineranno allo stesso momento
- [ ] le istanze dell'utente Z termineranno dopo l'istanza dell'utente X

0/3

Domanda 12 ◇ Il sistema operativo può essere visto come una **macchina estesa**. Quali aspetti rientrano in questa visione?

- [x] Il concetto di kernel
- [x] Il concetto di file e di directory
- [x] Il concetto di processo
- [x] Il concetto di thread

0/3

Domanda 13 ◇ La *schedulabilità* di un sistema realtime...

- [x] dipende dal tempo di esecuzione di ogni task
- [ ] concerne solo i task periodici
- [x] dipende dal numero di CPU/core
- [ ] dipende solo dal numero di task

3/3

Domanda 14 ◇ Subito dopo un **fork**, (quindi prima di un eventuale **exec**)...

- [x] il processo figlio può determinare il PID del padre
- [x] il processo padre e il processo figlio condividono lo stesso spazio di indirizzamento
- [ ] il PPID (Parent PID) del processo padre è uguale al PPID del figlio
- [ ] il PID del processo figlio è uguale a zero

2/3

Domanda 15 ◇ In un sistema con requisiti *hard realtime*...

- [ ] devo garantire tempi di risposta il più rapidi possibili
- [x] devo garantire la schedulabilità dei task
- [x] devo garantire tempi di risposta prevedibili
- [ ] devo garantire un elevato *turnaround time*

3/3

Domanda 16 ◇ L'algoritmo Shortest Remaining Time Next (SRTN)...

- [x] equivale a SJF se l'esecuzione dei processi non viene sospesa o prelazionata (preemption)
- [ ] può portare a una situazione di deadlock
- [ ] favorisce l'equità (*fairness*)
- [x] necessita di una stima del tempo totale di esecuzione dei processi

3/3

Domanda 17 ◇ Lo scheduling Multilevel Feedback Queue (MLFQ)...

- [x] è basato su priorità dinamiche
- [ ] è implementato utilizzando un albero binario bilanciato
- [ ] è una reimplementazione di CFS
- [ ] non supporta la preemption

3/3

Domanda 18 ◇ Il concetto di *locality*...

- [x] è importante per lo scheduling su sistemi con una CPU multi-core
- [ ] dipende dall'algoritmo di scheduling
- [ ] determina la priorità di scheduling di un processo/thread
- [x] è importante per lo scheduling su sistemi multi-processore

3/3

Domanda 19 ◇ Definisco una funzione *myhandler* come gestore del segnale *SIGINT*. Successivamente faccio un fork e poi *exec*. Se invio il segnale *SIGINT* ad entrambi i processi...

- [x] solo il processo padre eseguirà *myhandler*
- [ ] sia il processo padre che il processo figlio eseguiranno *myhandler*
- [ ] solo il processo figlio eseguirà *myhandler*
- [ ] né il processo padre né il processo figlio eseguiranno *myhandler*

3/3

Domanda 20 ◇ I processi di tipo *CPU-Bound*...

- [ ] non possono essere schedulati in un sistema con requisiti realtime
- [ ] vengono eseguiti con una priorità più alta rispetto agli altri processi
- [x] necessitano tipicamente di un algoritmo di scheduling pre-emptive
- [ ] effettuano numerose operazioni di input-output che bloccano la CPU

3/3

Domanda 21 ◇ Considera un sistema che utilizza l'algoritmo di scheduling denominato *Shortest Remaining Time Next* (SRTN) con *preemption*. Supponiamo che vengano schedulati i seguenti processi:

Processo	Momento di arrivo in coda	Tempo stimato di esecuzione
A	0	20
B	15	25
C	30	10
D	45	15

Quanto dovrà attendere il processo B prima di essere completato?

- [ ] 15
- [x] 55
- [ ] 5
- [ ] 40

3/3

Domande aperte [15 punti]

Rispondere ad ogni domanda in modo preciso, leggibile e conciso. Una risposta completamente corretta è valutata 3 punti.

Domanda 22 Qual'è la differenza tra scheduling *preemptive* e quello *non-preemptive*? Quali svantaggi ha lo scheduling *non-preemptive*?

☐∅ ☒∂ ☐∂ ☐√ 1/3

Nel non-preemptive sono i thread stessi che si passano la priorità; in caso di I/O si bloccano (bloccanti); Nel preemptive invece sono non bloccanti perché è un entità esterna (di solito kernel) che si occupa di schedulare (tramite interrupts) i thread

Domanda 23 Lo scheduling a coda singola nei sistemi multiprocessore è la soluzione ideale? Motiva la tua risposta.

☐∅ ☐∂ ☒∂ ☐√ 2/3

Ha i suoi vantaggi e svantaggi. Da un lato è un buon load balancing perché indirizza il thread dove è più libero però non è una soluzione scalabile ed essendo l'indirizzamento non sempre per forza sullo stesso core posso avere problemi di locality issues (perdo dei vantaggi). Non è la migliore.

Domanda 24 Quale problema cerca di risolvere lo scheduling di tipo *fair-share*?

☐∅ ☐∂ ☐∂ ☒√ 3/3

Nel caso io abbia per esempio 2 utenti contemporaneamente ma 1 con 20 processi e l'altro con solo 1, voglio far sì la CPU venga condivisa in modo equo a livello di utenti e non di numero di processi.

A B A B A B A B A B e non ~~A A A A A A B~~ ✓

Utente 1: A Utente 2: B

Domanda 25 In che modo lo scheduling tradizionale Unix favorisce i processi I/O Bound rispetto a quelli CPU Bound?

☒∅ ☐∂ ☐∂ ☐√ 0/3

Essendo lo scheduling dinamico e ma soprattutto round-robin io do comunque la possibilità a tutti di eseguire, questo può portare problemi a lunghi processi che hanno bisogno di tanta CPU essendo interrotti spesso mentre nei processi I/O BOUND lo non è un problema perché attendo le risorse durante

Domanda 26 Come funziona lo scheduling di tipo *round-robin*? Quale parametro è necessario per il suo funzionamento? Quali fattori influiscono sulla scelta di un valore per questo parametro?

☐∅ ☒∂ ∂ ☐√ 1/3

Lo scheduling round robin funziona tramite ~~liste~~ circolari poste a livelli di priorità diversi. Ad ogni processo è assegnato un quantum di tempo in cui può eseguire. Quando termina, la palla passa al processo successivo nella stessa lista. Si parte dalla lista più alta e si scende quando tutti i proc. di quella lista terminano

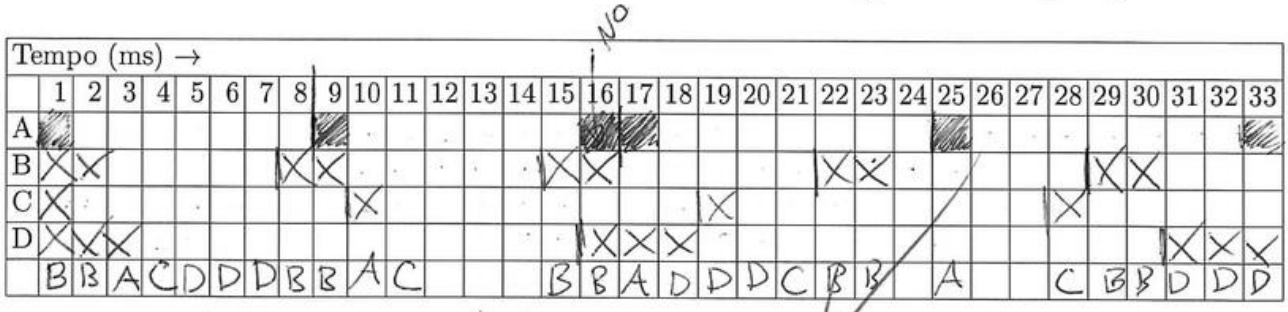
Scheduling realtime [12 punti]

Domanda 27 Considera un sistema realtime con i seguenti 4 task periodici:

- A, tempo di esecuzione 1, periodicità 8 — 2°
- B, tempo di esecuzione 2, periodicità 7 — 1°
- C, tempo di esecuzione 1, periodicità 9 — 3°
- D, tempo di esecuzione 3, periodicità 15 — 4°

Scegli l'algoritmo **più semplice/opportuno** tra i due visti durante il corso (RMA o EDF).
► **Indica chiaramente l'algoritmo scelto sulla riga tratteggiata sotto alla tabella e motiva la tua scelta. [1 punto]**
► **Completa la tabella indicando l'ordine di scheduling in base all'algoritmo scelto.** *Nota: considera che si tratta di un sistema non-preemptive con una sola CPU/un solo core.* **[3 punti]**

B A C D

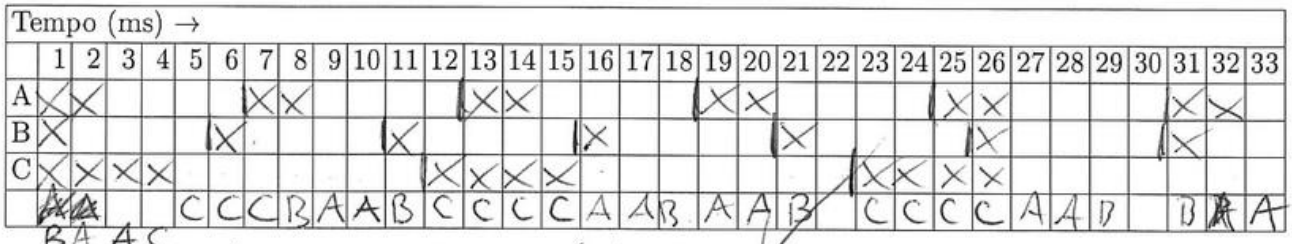


RMA (U = 0,73) ≤ (n(2^{1/n} − 1) = 0,75) vedi foglio ☐w ☐p- ☐p+ ☒c 4/4

Domanda 28 Considera un sistema realtime con i seguenti 3 task periodici:

- A, tempo di esecuzione 2, periodicità 6 — 2°
- B, tempo di esecuzione 1, periodicità 5 — 1°
- C, tempo di esecuzione 4, periodicità 11 — 3°

Scegli l'algoritmo **più semplice/opportuno** tra i due visti durante il corso (RMA o EDF).
► **Indica chiaramente l'algoritmo scelto sulla riga tratteggiata sotto alla tabella e motiva la tua scelta. [1 punto]**
► **Completa la tabella indicando l'ordine di scheduling in base all'algoritmo scelto.** *Nota: considera che si tratta di un sistema non-preemptive con una sola CPU/un solo core.* **[3 punti]**



EDF (U = 0,85) > 3(2^{1/3} − 1) = 0,78 vedi foglio ☐w ☐p- ☐p+ ☒c 4/4

+12/8/25+

Domanda 29 Considera un sistema realtime con i seguenti 2 task periodici:

- A, tempo di esecuzione 3, periodicità 7 2°
- B, tempo di esecuzione 1, periodicità 6 1°

Scegli l'algoritmo **più semplice/opportuno** tra i due visti durante il corso (RMA o EDF).
► **Indica chiaramente l'algoritmo scelto sulla riga tratteggiata sotto alla tabella e motiva la tua scelta. [1 punto]**
► **Completa la tabella indicando l'ordine di scheduling in base all'algoritmo scelto.** *Nota: considera che si tratta di un sistema non-preemptive con una sola CPU/un solo core.* **[3 punti]**

Tempo (ms) →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
A	X	X	X	·			X	X	X	·	·	·	·		X	X	X					X	X	X	·				X	X	X		
B	X						X						X						X						X						X		
	B	B	A	A			B	A	A	A			B		A	A	A		B			A	A	A	B				A	A	A	B	

BA

B RMA (U ≤ 0,55) < (2(2^1/2 − 1) ≈ 0,82) ☐w ☐p- ☐p+ ☒c 4/4

vedi foglio

=