Politecnico di Milano – Sede di Cremona
Anno Accademico 2016/2017

Architettura dei Calcolatori e Sistemi Operativi

Esame – 27.07.2017

Prof. Carlo Brandolese

Cognome _____ **Nome** _____

Matricola _____ **Firma** _____

Istruzioni

1. Scrivere con cura, negli spazi sopra segnati, il proprio cognome, nome, numero di matricola e apporre la firma.
2. È vietato consultare libri, eserciziari, appu nti ed utilizzare la calcolatrice e qualunque strumento elettronico (inclusi i cellulari), pena l'invalidazione del compito.
3. Il testo, debitamente compilato, deve essere riconsegnato in ogni caso.
4. Il tempo della prova è di 3 ore

Valutazione

Domanda	Voto	Note
A		
B		
C		
D		
E		
F		

Domanda A

Si implementi in linguaggio assembly ed in forma ricorsiva l'algoritmo di Euclide per il calcolo del massimo comun divisore. Si ricorda che tale algoritmo è descritto in forma ricorsiva dalle seguenti equazioni:

$$\gcd(a,b) = \begin{cases} a & se\ b = 0 \\ \gcd(b, a \bmod b) & se\ b \neq 0 \end{cases}$$

Si tenga presente che il prototipo della funzione deve essere il seguente:

```
int gcd( int a, int b );
```

Si ricorda che il linguaggio assembly del MIPS dispone delle seguenti istruzioni:

```
DIV $r1, $r2      => $lo = $r1 / $r2,      $hi = $r1 % $r2

MFHI $r1          => $r1 = $hi

MFLO $r1          => $r1 = $lo
```

Domanda B

Scrivere un programma C che, attraverso la creazione di 3 thread, determini quali numeri da 1 a 3000 sono numeri primi. Infine tutti i numeri primi devono essere stampati a video in ordine crescente.

Domanda C

Si consideri un sistema con uno spazio di indirizzamento di 1GByte e dotato di una cache dati ed una cache istruzioni disgiunte. La cache dati (DCACHE) è una cache completamente associativa della dimensione di 32Kbyte con una linea di 256 byte, mentre la cache istruzioni (ICACHE) è una cache direct-mapped della dimensione di 128Kbyte con una linea di 512byte.

Sulla base di queste informazioni si indichi la struttura dell'indirizzo visto dalle cache, descrivendo i vari campi e il loro significato.

DCACHE

ICACHE

Sapendo che:

- Il tempo di accesso alla cache in caso di hit è di 1 ns
- L'accesso alla memoria RAM avviene a parole di 32 bit
- Il tempo di accesso alla RAM in modalità normale è di 40 ns
- Il tempo di accesso alla RAM in modalità burst è di 50 ns per la prima parola e 10 ns per le parole successive
- L'hit rate della DCACHE è pari al 90%, mentre l'hit rate della ICACHE è pari al 95%

Si calcolino i tempi medi di accesso alle due cache.

T_{DCACHE}

T_{ICACHE}

Si consideri quindi l'esecuzione di un programma in cui:

1. Il 50% delle istruzioni non accede alla memoria
2. Il 40% delle istruzioni richiede un accesso alla memoria
3. Il 10% delle istruzioni richiede due accessi alla memoria

Si calcoli il tempo medio di esecuzione di una istruzione per tale programma, supponendo che in assenza di stalli tutte le istruzioni siano eseguite in 1 ciclo.

T_{AVE}

Si calcoli infine il miglioramento delle prestazioni per il programma in esame rispetto all'esecuzione sullo stesso sistema ma in assenza di memoria cache.

Domanda D

Si consideri il seguente insieme di processi:

Process	Arrival Time (T_A)	Execution Time (T_E)
P1	0	10
P2	0	2
P3	3	8
P4	5	5
P5	8	1

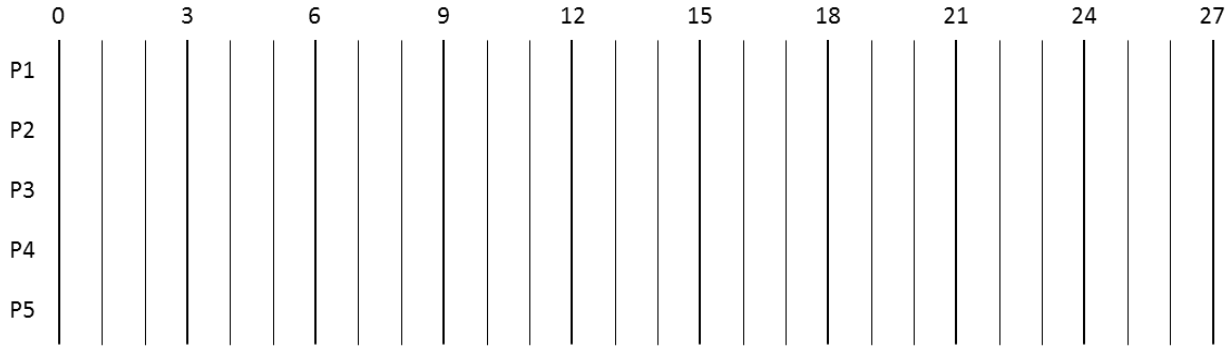
Si esegua lo scheduling di tali processi secondo i due seguenti algoritmi:

- Shortest Job First, non-preemptive
- Shortest Remaining Time

Per ognuno dei due casi, quindi, si svolgano i seguenti punti:

- Si indichi il tempo reale di esecuzione di ogni processo
- Si calcoli il tempo di attesa medio T_W dei processi
- Si calcoli il tempo di attesa massimo T_M dei processi
- Supponendo che un processo P_n abbia una deadline di completamento $T_{D,n}$ data dalla seguente relazione $T_{D,n} = T_{A,n} + 1.5 * T_{E,n}$, si indichino quali processi sono completati entro la deadline in ognuno dei due casi.

Shortest Job First



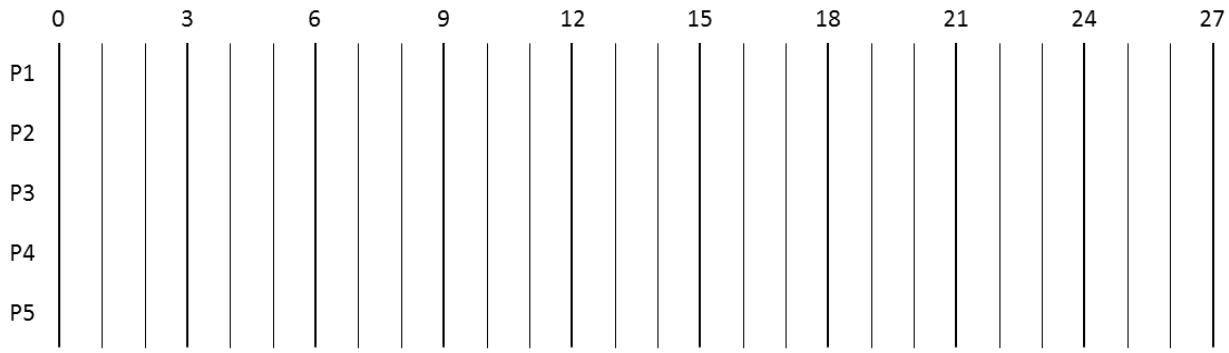
T_W =

T_M =

Processi completati entro la deadline:

Processo	Tempo Reale di esecuzione	Rispetto della deadline
P1		
P2		
P3		
P4		
P5		

Shortest Remaining Time



T_W =

T_M =

Processi completati entro la deadline:

Processo	Tempo Reale di esecuzione	Rispetto della deadline
P1		
P2		
P3		
P4		
P5		

Domanda E

Si consideri un calcolatore dotato di sistema operativo Linux e in cui valgono le seguenti specifiche:

- Le dimensioni dei blocchi sono di 512 byte
- Per l'apertura dei file è sempre necessario accedere a:
  - I-node di ogni cartella o file presente nel percorso
  - Blocco per il contenuto di ogni cartella presente nel percorso
  - Primo blocco dati del file

Dato il contenuto del seguente volume:

I-lista:	<0,dir,10> <1,dir,11> <2,dir,12> <3,norm,{100,101,102,103}> <4,dir,13> <5,norm,{200,201,202,203,204,205}> <6,dir,14> <7,dir,15> …
Blocco 10:	… <1,home> <2,bin> <4,tmp>
Blocco 11:	… <8,rl> <6,acso> <9,fondamenti>
Blocco 12:	… <10,bash> …
Blocco 13:	… <11,var> …
Blocco 14:	… <7,files> <12,tde> <13,solutions>
Blocco 15:	… <14,paperino.dat> <3,pluto.dat> <5,pippo.dat> …

1. Per ciascuna delle chiamate di sistema sotto riportate, si indichi la sequenza di accessi agli I-Node e ai blocchi (del tipo: I-Node X oppure Blocco Y), indicando inoltre se gli i-Node o i blocchi sono in memoria o su disco (M o D). Si assumano le 4 chiamate sequenziali.

Chiamata di sistema	Sequenza di accessi
fd = **open**(“/home/acso/files/pippo.dat”, O_RDWR)	
lseek(fd, 2049, SEEK_SET)	
read(fd, buffer, 10)	

2. Assumendo di creare un nuovo file di dimensione 2KB con la seguente istruzione:

fd2 = **open**(“/home/acso/files/topolino.dat”, O_RDWR | O_CREAT, S_IRUSR | S_IRGRP | S_IROTH)

Riportare qui sotto le modifiche effettuate al contenuto del volume dopo la creazione del file stesso.

Inoltre data la struttura dei blocchi liberi seguente specificare quali blocchi saranno occupati dal file *topolino.dat* utilizzando le politiche di allocazione:

Contigua – First Fit

Contigua – Best Fit

Contigua – Worst Fit

Spazio fisico disponibile

Blocco	Stato
…	
300	
301	
302	
303	
304	
…	
400	
401	
402	
403	
…	
500	
501	
502	
503	
504	
505	
…	

Domanda F

Si descriva nel modo più preciso e chiaro possibile il meccanismo di register forwarding utilizzato nell'architettura MIPS, indicando in particolare quale tipo di problema risolve questo meccanismo e quali stadi della pipeline sono coinvolti.