

Sistemi di Calcolo (A.A. 2025-2026)

Corso di Laurea in Ingegneria Informatica e Automatica
Sapienza Università di Roma



Compito (22/07/2026) – Durata 1h 30'

Inserire nome, cognome e matricola nel file `studente.txt`.

ISTRUZIONI PER STUDENTI DSA: svolgere a scelta due parti su tre.

Parte 1 (programmazione IA32)

Nel campo della security è fondamentale poter filtrare rapidamente gli eventi in base alla loro criticità. A tale scopo, è utile disporre di un meccanismo in grado di assegnare uno score di rischio a ciascun evento e di calcolare un indicatore complessivo del rischio associato a una sequenza di eventi. In questo esercizio, si chiede di tradurre in assembly una funzione che, dato un insieme `events` di eventi identificati da valori `short`, la sua dimensione `n` e due soglie `low_th` e `high_th`, calcola il fattore di rischio totale. Ogni evento contribuisce al totale in base allo score restituito dalla funzione `risk_score` e al superamento delle soglie indicate. Nella directory E1, si traduca in assembly IA32 la seguente funzione C scrivendo un modulo `e1A.s`:

```
int suspicious_events_score(short* events, int n, int low_th, int
high_th)
{
    int total = 0;
    int score;
    short* end = events + n;

    while (events < end) {
        score = risk_score((int)*events);
        if (score > low_th) {
            total += 1;
        }
        if (score > high_th) {
            total += 1;
        }
        events++;
    }
    return total;
}
```

L'unico criterio di valutazione è la correttezza. Generare un file eseguibile `e1A` con `gcc -m32 -g`. Per i test, compilare il programma insieme al programma di prova `e1A_main.c`.

Non modificare in alcun modo `e1A_main.c`. Prima di tradurre il programma in IA32 si suggerisce di scrivere nel file `e1A_eq.c` una versione C equivalente più vicina all'assembly.

Parte 2 (programmazione di sistema POSIX)

Si scriva una funzione multi-thread che verifichi la simmetria di un array rispetto al suo centro (array palindromo). Ad esempio:

[1, 2, 3, 4, 3, 4, 1] è simmetrico [1, 2, 3, 4] non è simmetrico in posizione 0

Nello specifico, si scriva nel file `E2/e2A.c` una funzione con il seguente prototipo:

```
int check_symmetry(const int * input, const unsigned short length)
```

che, dato un array di interi `input` di lunghezza `length` restituisce `-length-1` se l'array è simmetrico rispetto al suo centro, oppure l'indice della prima posizione per cui non è valida la proprietà di simmetria. Un array vuoto è per definizione simmetrico.

La soluzione deve fare uso di thread, ognuno dei quali deve confrontare una posizione di indice `i` con la rispettiva posizione di indice `length-i-1`. **Attenzione:** Nel caso in cui l'array non fosse simmetrico la ricerca della posizione va calcolata successivamente senza fare uso di thread, al fine di evitare problemi di concorrenza.

Per i test, compilare il programma usando lo switch `-lpthread` insieme al programma di prova `e2A_main.c` fornito, che **non** deve essere modificato.

Parte 3 (quiz)

Si risponda ai seguenti quiz, inserendo le risposte (A, B, C, D o E per ogni domanda) nel file `e3A.txt`. Una sola risposta è quella giusta. Rispondere E equivale a non rispondere (0 punti).

Domanda 1 (Memoria virtuale)

Si consideri un sistema di memoria virtuale con spazio di indirizzi (logico e fisico) a 24 bit e pagine da 256 KB. Data la seguente tabella delle pagine (indice pagina → frame, valori esadecimali), a quali indirizzi fisici corrispondono gli indirizzi logici `0x15C3F0`, `0xAAA0B4`, `0xFCFFF0`?

idx	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x0	0A	25	3C	18	2F	11	33	1C	00	2A	15	38	03	21	3E	0D
0x1	30	1F	26	0B	39	14	2D	08	35	02	1A	3B	27	10	2E	05
0x2	3A	0F	22	31	1D	06	28	13	3F	24	07	19	29	01	36	0E
0x3	23	0C	37	1E	04	2B	32	17	09	3D	20	16	34	12	1B	2C

A	<code>0x45C3F0</code> , <code>0x1EA0B4</code> , <code>0xB0FFF0</code>	B	<code>0x45C3F0</code> , <code>0x1EA0B4</code> , <code>0xB4FFF0</code>
C	<code>0x51C3F0</code> , <code>0x1CA0B4</code> , <code>0xB0FFF0</code>	D	<code>0x44C3F0</code> , <code>0x9EA0B4</code> , <code>0xBCFFF0</code>

Motivare la risposta nel file `M1.txt`. **Risposte non motivate saranno considerate nulle.**

Domanda 2 (Ottimizzazione)

Considera il seguente frammento di codice in C.

```
for (int i = 0; i < N; i++) {  
    A[i] = B[i] + (x * y);  
}
```

Quale trasformazione applicherà il compilatore se si attiva l'ottimizzazione di tipo Loop-Invariant Code Motion (noto anche come Hoisting)?

A	Il ciclo viene duplicato per dimezzare le condizioni di controllo	B	L'espressione <code>x * y</code> viene calcolata una sola volta prima dell'inizio del ciclo e il suo risultato memorizzato in una variabile temporanea
----------	---	----------	--

C	La moltiplicazione $x * y$ viene sostituita da una serie di somme consecutive	D	L'intero ciclo viene eliminato dal compilatore perché considerato "dead code"
----------	---	----------	---

Motivare la risposta nel file M2.txt. **Risposte non motivate saranno considerate nulle.**

Domanda 3 (Pipelining)

Si consideri la seguente sequenza di istruzioni che vengono eseguite da una CPU con pipeline a 5 stadi (Fetch, Decode, Execute, Memory, Write-Back)

```
1: subl $8, %esp
2: addl %eax, %esp
3: movl $0x41414141, %(esp)
4: addl $4, %eax
5: movl $0x42424242, %(eax)
6: addl $8, %esp
```

Quanti cicli di clock vengono richiesti per completare tutte le istruzioni assumendo che gli hazard vengano risolti con stalli?

A	13	B	19
C	16	D	17

Motivare la risposta nel file M3.txt. **Risposte non motivate saranno considerate nulle.**

Domanda 4 (Allocazione di memoria)

Si consideri un semplice allocatore di memoria dove, potendo scegliere, la `malloc` riusa il blocco libero con l'indirizzo più basso. Assumere per semplicità che non vi sia alcun header e che i blocchi allocati vengono arrotondati a una dimensione multiplo di 4 byte. Inoltre, l'allocatore non divide blocchi liberi in più blocchi di dimensione inferiore, né fonde eventuali blocchi liberi adiacenti in un blocco di dimensioni superiori. Qual è il contenuto dell'heap alla fine della seguente sequenza di operazioni ("X" denota 4 byte allocati e "." denota 4 byte liberi)?:

```
p1 = malloc(5)
p2 = malloc(12)
p3 = malloc(4)
p4 = malloc(9)
free(p1)
free(p3)
p5 = malloc(8)
p6 = malloc(13)
```

A	.. XXX . XXX XX XXXX	B	XX XXX . XXX XXXX
C	XX XXX . XX XXXX	D	Nessuna delle precedenti

Motivare la risposta nel file M4.txt. **Risposte non motivate saranno considerate nulle.**